

Latex Made Easy

Darshil Patel
Computer Science
North Carolina State
University
Raleigh, North Carolina
dpatel14@ncsu.edu

Muppidi Harshavardhan
Reddy
Computer Science
North Carolina State
University
Raleigh, North Carolina
mreddy2@ncsu.edu

Ruthvik Mandava
Computer Science
North Carolina State
University
Raleigh, North Carolina
rmandav@ncsu.edu

Srinivas Parasa
Computer Science
North Carolina State
University
Raleigh, North Carolina
sparasa@ncsu.edu

ABSTRACT

With the advancement of technology day by day, the level of comfort for human beings interacting with anything keeps increasing. In the recent time one such area is formatting technical research papers or be it any document. Generally writing a technical paper is very important to scholars, graduates and also professors in all universities irrespective of the field of study. Since it adds value to their research and helps them reaching out to wide range of audiences. While various editors in the market have their own advantages, disadvantages, and complications, Latex[1] is known for it's high-quality setting and non-word processing system which assures people not to worry about the appearance but concentrate on the content. But we observed that it still requires people to learn its syntax to come up with a perfectly formatted document. There are many tools in the market that provides hundreds of templates to start with, but nowhere could we find a proper interactive GUI that just asks for only the content and then builds the code independently, without the user actually studying all the syntax needed to format. Therefore the aim of this project is to develop a web application that generates the required code for the selected structures by the user by just taking just the context as the input. This paper discusses the purpose, previous applications generally used, the implementation of our project and an evaluation of the application by users based on their experience using it.

Keywords

Web Application; L^AT_EX; paper; template; Software Engineering; OverLeaf; ShareLatex

1. INTRODUCTION

Increasing need and importance of documents has adversely made people to focus on the format of the documents which needed uniformity with proper guidelines for various types of documents. Especially when it comes to typesetting journal articles, technical reports, books, and slide presentations or modularization of large documents comprising of many sections, tables, figures, cross-references, complex for-

mulae and mathematical equations, automatic generation of glossary and indexes, easy font settings and control of the font over many sections in the complete document, LaTeX outworks all of the prominent editors. So basically latex is a helpful document processing system which is very popular not only in the field of Computer Science but all other technical fields which requires the students/mentors to write technical papers for their research work. Even though, professionals related to the field of Computer Science might find writing papers in Latex comparatively easier, there is a large amount of students from other majors, for whom documenting their research work feels rather tough. Presently in the market these LaTeX editors are quite complicated and needs users to understand the syntax to actually format it. This again increases the difficulty for any user thinking to use LaTeX and since it's superiority surpasses the effort needed to format, people still opt to edit and format their documents using LaTeX. So now what if there was an interface which makes people easier to format and get a formatted document just by giving the content. To proceed with this we weren't sure as to how much users would actually like a interface for this purpose.

With this idea we wanted to know what users actually felt about the usage of latex, in terms of time consumed, the effort needed and how complex is it to format a document. For this study we have gathered responses from students in various technical backgrounds who use LaTeX to edit their technical papers or any kind of documents. We have enquired as to how confidently users are able to format their documents using LaTeX and how complicated it is and at the same time how much time consuming it is to format the documents. Finally we also gathered data, whether they would like to have a web application which does the formatting them by only giving the content to be included in the paper.

From the survey study we discovered that the users were actually having many difficulties in understanding the syntax and formatting the documents and also apart from this we got to know that this formatting was actually consuming a lot of their time which they could actually invest in their research. So this assured us that our assumptions about

how users are facing issues using LaTeX and whether the users would like a web application serving this purpose was cleared. Now with assurance based on the survey and considering the various factors like complexity and time we have to build our application and at the same time addressing the issues majorly faced by users first, given time constraint we decided to address secondary issues.

So firstly the ease and comfort was our primary concern which adversely implies saving large amounts of time. So we decided to initially come up with a certain fixed format as of now which students from various technical backgrounds can use to develop their research paper since a majority percentage of users are those who are looking to publish their research work and document their projects. Therefore this particular format will cater especially to all such users. This format will basically consist of options to enter title, author college, degree and date when a user first logs in, but none of these entries are compulsory because a user might not need the latex code version for all the elements of document. Following this we have dedication, acknowledgement, abstract entries to fill. Finally a page where users can add chapters and subsections under them and at the same time add tables or images and caption them according to their need. With all this done a user can either save his/her work and return back to continue right where he left off, or else they could just download the latex code which gets downloaded in a zip format.

Throughout this report we will covering various aspects with respect to the implementation of the complete application, various technologies used to achieve this and at the same what was the user experience with the application as well.

2. SYNOPSIS

2.1 Previous Work

Our main motive behind this project is to make formatting documents much easier and comfortable and in fact the various editors presently in the market were one of the main reasons which encouraged us to create this web application. Our Research over all the various editors in the market gave us a great insight of the various issues. In general the major applications used by various users were Microsoft Word, Google Docs, ShareLatex, Overleaf, Texmaker, Texstudio, Texworks and so on. With this knowledge we started to uncover what were the major issues while using these applications for formatting documents and at the same time the pros of it. Starting with Microsoft word which is one if the most widely used application for editing documents but when it comes to official documents at the same time research works, with the complexity of formats involved in it, users preferred using the LaTeX(Lamport Tex). When it comes to basic editing using Microsoft Word would suffice which is not the case.

So Basically LaTeX is a document preparation system, which is different from Microsoft Word and other similar applications where we need to concentrate on formatting our content too while latex comes to aid which helps in formatting the document automatically with predefined markup tagging conventions. LaTeX uses the TeX typesetting program for formatting its output, and is itself written in the TeX macro language. As the complexity increased people moved to LaTeX editors since many applications were try-

ing to make this task easier with their own interfaces and features. On closer observation of other editors like Texmaker which is fully unicode and supports a large variety of documents. Some of it's features include code folding, code completion, fast navigation, 'Master Mode', error handling, latex documentation, regular expressions support, but all of these as we know needs the user to have knowledge of the basic syntax to format and in case when users specifically would like specialized formatting it becomes difficult for them study the syntax with respect to only such specific element and then incorporate it in their document. This is one of the major issue we observed though it has a lot of advantages when compared to basic editing applications like microsoft word but still lacks adaptability in terms of learning the syntax. We have also discovered there were few performance issues with respect to Texmaker, for example when there was large document with many sub documents, in that case changing even just the label, chapter or section would incur a great deal of overhead with respect to the performance time.

Then coming to Texstudio which has been actually forked from Texmaker in 2009, because of the non-open development process of Texmaker and due to the different philosophies concerning configurability and features. TexStudio was pretty much similar except for few added features like table formatting, link overlay, drag and drop support. There were few problems with this too, like highlighting of the invalid/unknown LaTeX command doesn't work well. Besides this, our main concern was with the users understanding the syntax. Down the line with further research we came across Sharelatex which is an online Latex real-time editor, compiler which removes people from the havoc of downloading required packages to run and compile Latex documents. Yet again, user needs to learn the tex syntax in order to write a document in ShareLatex which is a very cumbersome problem for people from non-technical fields. A similar application was overleaf which supports almost everything as sharelatex but just varies in terms of business aspects and free available features.

After looking over all the various available editors we understood that these tools majorly concentrated in providing the templates and then asking the user to edit the content in the code to have their own document ready. But there might be many instances where a user just needs a specific content to be inserted into his/her document and needs specific instructions in hand rather than going through hundreds of templates and search the required code. Latex-Made-Easy provides many options to users to select and asks people for only the content they want to keep in specific sections and generates the code independently thereby reducing the burden of searching for the correct code section in templates and editing the same and compiling the same.

2.2 Software Development Cycle

For proper execution of the complete project starting with gathering required data, design, implementation and testing, an efficient SDLC has to be selected based on the project expectations. A Software Development Life Cycle is a conceptual model used in project management that describes the stages involved in an information system development project, from an initial feasibility study through maintenance of the completed application.

Therefore in our case we have decided to go with Agile

Methodology since agile being an incremental software development methodology we have the liberty to keep developing based on our outputs from time to time.

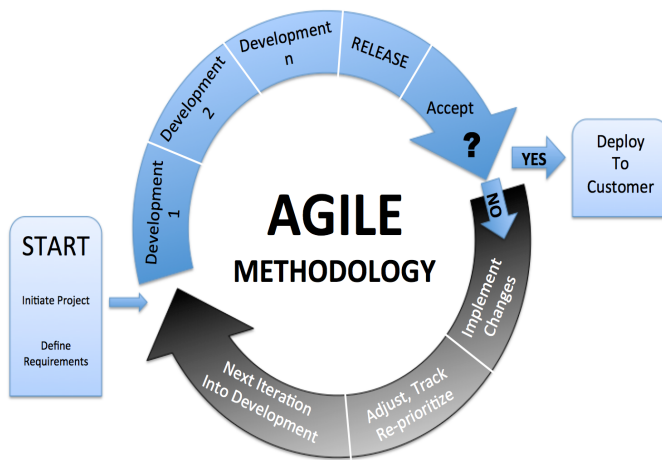


Figure 1: Development Life Cycle

Along with this we have decided to go with TLD(Test Last Development), since the scale of the application as of now is small and therefore given the time constraint we wanted to invest more time for development and if we would have gone with TFD we might end up making the development process complicated and time may have been wasted. In that case, hence to be on the safer side we went with TLD. For testing we have decided to use Jasmine and Karma testing frameworks.

2.3 Initial Survey

Based on the decided project scope, we conducted a survey using Google Forms to determine the need and user's expectations for the project. The questions focused more on use of latex, time taken for writing documents in latex and the need of such an application which can eradicate the requirement to learn latex and help technical writers to devote their time more on working with their content and research as compared to the time invested in formatting their work in form of reports and documents.

From this initial survey we have gathered the required data based on which we could focus on prioritizing the features to be developed first and then in an iterative fashion keep adding features in the given time frame and ensure a working product at the end.

2.4 Design and Implementation

For the implementation of this web application we have decided to implement backend using Node.js. With the backend support from node.js the frontend was developed using angular.js. For the database support of this application we have decided to go with a NoSQL database, mongodb. The combined design implementation of these technologies is straight forward as the following diagram below.

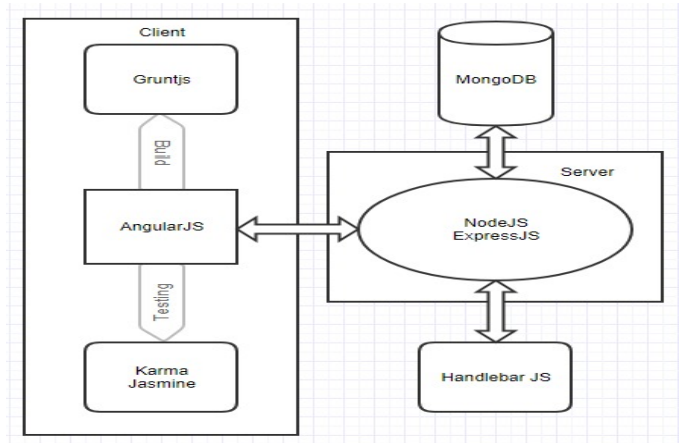


Figure 2: Design of Application

A user through the interface sends in various kinds of requests be it login or navigating to the next page or to saving his work or getting the latex all these requests are sent to the node.js server. Now on the backend the parsing of this content into latex is supported by handle bars along with node.js on the backend. Over here the REST API will be created using Node.js. This API will be communicating with the database to store the user information and the saved modules. One of the endpoints will also be responsible for creating the latex code based on the user input provided at the front end. The latex code will also be saved in the database with the user info so that the user can login and fetch their saved modules again. This being said about the basic outline implementation of the web application there are multiple factors and reasons why we particularly selected these technologies and the design for this implementation.

The reason behind specifically going with Node.js is because it has an inbuilt support of the npm package manager which provides a list of packages which are useful throughout web development using MEAN stack. Also following the MEAN stack means having to do with just one programming language throughout the application. It is very easy to deploy Node.js application on platforms like heroku and AWS with absolutely no cost. It operates on a single thread for processing HTTP requests using non blocking calls for handling multiple requests. Moreover node is very fast as compared to some other servers like Apache. Also, as npm is an open source package manager and provides tons of helpful packages, node.js becomes the first choice while developing such applications.

While for the database we went with MongoDB which is a NoSQL document database which uses JSON like documents with schema. There are many reasons why we choose MongoDB as database to our development stack. One primary being MEAN stack which is very popular in open source community. Since the community is large we can get solutions easily for any problems we face throughout the project.

The problem with not using SQL database is the user in our application can actually skip some sections which makes the backend data unstructured. SQL is not at all good for unstructured data because space is wasted for the empty rows in the column which is a huge issue as the system scales. Also since the application stores minimal amount

of data in a session like only login info and latex json. Designing database schema would become a cumbersome task and takes considerable amount of time.

Considering the benefits we have with MongoDB and drawbacks that SQL has specific to our application we moved forward with MongoDB. But as we went through the project we found out that MongoDB perfectly aligned with our primary purpose. We are trying to construct a json from all the data the user has entered and sending the data to backend to convert it into latex code using handlebars[11]. During this process we always passed data in json format. As JSON can be easily stored in MongoDB without any formatting required at backend.

For the front end development we have Angularjs which helps to build the applications very fast. It reduces the development time as it allows the developers to use directives which are basically reusable HTML code. If there is a set of HTML tags which have to be used at multiple places there is no need to use the tags at multiple places, instead just create a directive containing those tags and use the directive everywhere.

Also, Angularjs allows real time DOM manipulation which was an essence to the project as we needed to add multiple DOM elements as per the users selection. For example, if the user wants to add a new Section, that is when we have to provide a textarea to the user for adding the Section, which can be known only at runtime.

Moreover, Angularjs also allows us to use build tools like gruntjs and gulpjs to build and bundle the whole Angularjs code into one. We have used gruntjs for bundling all the HTML and Javascript files into one file for execution. We do not need to link newly created javascript files into the index file because the gruntjs build takes care of that.

Angularjs also supports multiple linting tools like jscs and jshint which helps to keep the code clean and readable. While using gruntjs for the build, we can actually check the code for any linting errors. Linting errors basically means the code formatting errors.

Along with all such things which are important to create a good software, it also supports testing frameworks like Jasmine and Karma. This allowed us to write test cases for the code we developed. The current testing coverage is 90% which means that 90% of the lines in the whole code base is being tested by the number of test cases written using Jasmine and Karma.

Thus, using Angularjs was essential not only for developing, but also for testing, linting, building as well as communication of the application to the server.

With the support of backend and frontend being developed with angularjs the user's request from the front end is received as a JSON object with predefined keys. There are a lot of components that these keys refer to such as a title (Title Page), table (a Table), a list (Unordered or Ordered List), a chapter (Chapter), subsection (a subsection added into the chapter a level above).

To make code clear and modularized, each component is to be parsed and changed to Latex code and added to the main template. As well processing a key and its values iteratively and imbibing them inside the latex tags requires a lot of codes of line. It also increases the burden of checking for all the necessary exceptions that could make latex code doesn't work. For example, suppose user gives you the content of a chapter with two subsections where one subsection

has a table and the other has a list. So the design would become complex to parse and check for a type of element, then retrieve the latex code for the type and put the content in between the tags and append it to the main code.

Therefore final objective is to modularize the processing each element and also remove the burden of processing the content and adding into their respective positions of latex tags.

Handlebars provide the power necessary to build semantic templates effectively with no effort. It allows templates to be precompiled and included as javascript code. Take acknowledgment as a simple example. Frond End delivers an JSON object with a key that refers to the type 'Acknowledgement' and we just retrieve just the object and doesn't process the content. We call respective handlebar for Acknowledgement which looks like below with the content and compile them together that imbibes the content into the respective positions;

```
Content: {acknowledge}
Handlebar for Acknowledgement:
{{#if acknowledge}}
    \begin{acknowledgements}
        {{ acknowledge }}
    \end{acknowledgements}
{{/if}}
```

This way it helped us to modularize it for every type to get processed by their respective handlebars and then get the rendered latex code as string which can be appended to the main latex code that is being generated while iterating through all the elements in the JSON.

Performance is also enhanced by using our precompiled Handlebars.js templates as it was faster than actually parsing the content and adding it to the latex components. Handlebars also provide block expressions to iterate into json values to any depth. It provides all loop statements and if/else condition checks to see if there exists a key and then adds it to the latex component. There are many components that are optional. So handlebars expression checks for the presence and then adds it to the latex parts if present.

2.5 Continuous Integration and Coveralls

Continuous Integration is a software engineering process which requires the developer to frequently submit the code into the shared repository several times. Each check-in is verified by a continuous integration automated build system like Jenkins and Travis CI[12] which allows the developers to detect issues very fast. Due to this reason, we used the Travis CI automated build system to continuously build our code as and when the commits are made.

The Travis CI build has been configured with grunt to run tasks for the following things:

1. Build - This task is used while doing local development. The task takes care of minifying (uglifying¹)

¹Minimizing the Javascript, HTML and CSS files so that people can not understand the code when they try to view the source. This is majorly for the security purposes because the readable client side code can very easily display passwords if the smart user keeps proper breakpoints

2. Test - This task kicks off the testing framework. The EcmaScript language specification used with the AngularJS for this project development is EcmaScript6(ES6) which still can not be compiled by many task runners like grunt and many testing frameworks like Karma and Jasmine which are also used in this project for testing. ES6 has its own major positives of being used for the project². This added an additional step to transpile the ES6 code to the language specification which can be understood by all these build and test tools. The task takes care of transpiling the code prior to testing the code and it also takes coverage into consideration.
3. Default - The default task contains all the steps which are required for making the code deploy ready. It tests the code for linting errors³ which is beneficial to keep the code quality and code style uniform even though there are multiple developers. It also runs the above mentioned Build as well as Test task and also sends the coverage report to Coveralls.io[13]

Travis CI runs the above mentioned default task while a pull request is raised, merged or even if the code is pushed directly to the master branch. This is to make sure that the code in the master branch at any point in time is completely error free.

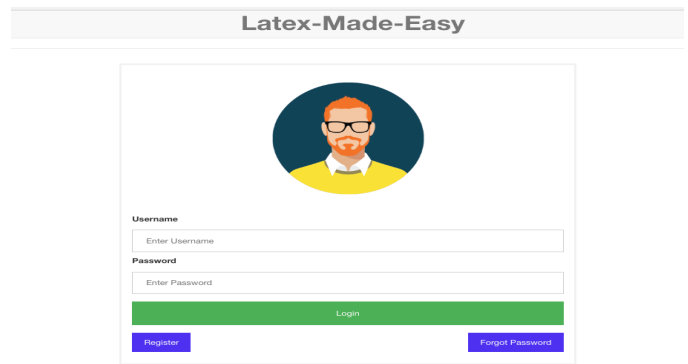
As mentioned above, the Travis CI sends the test coverage to Coveralls.io which in turn shows all the test case coverage statistics and trends. It helps to embed the coverage report as well as the build status report to the README.md file in real time. If a particular Travis CI build fails, the README.md file of the repository gets automatically updated to show that the current status of the build is failing. This helps all the developers across the board to know the real time status and test coverage of the code.

2.6 Functionalities

Since our application is web based we gave instructions for the user to install the application in github ReadMe. The application can be run locally by the user. These are very simple commands that can be run by the user in limited time. Before using the application, the user has to register on the website which would help us to maintain sessions. After registration the user can login with his credentials. Different validations and checks are incorporated on the sign in page like forget password which sends the email to the registered email with the password the user gave during registration. The visual diagram of the registration and login page is shown below.

²<https://coherent-labs.com/es6-standard-with-babel-js/>

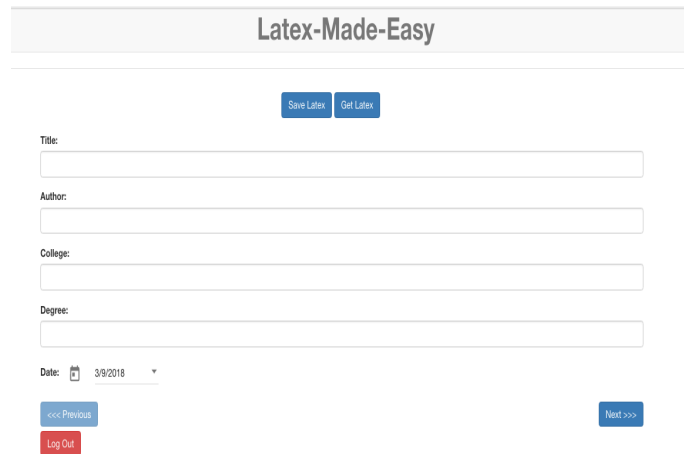
³Testing the code for syntax errors and code style errors.



The image shows the 'User Login Page' for 'Latex-Made-Easy'. At the top, there's a header with the text 'Latex-Made-Easy'. Below the header, there's a large circular profile picture of a man with glasses and a yellow shirt. Underneath the profile picture, there are two input fields: 'Username' with the placeholder text 'Enter Username' and 'Password' with the placeholder text 'Enter Password'. Below these fields is a green 'Login' button. At the bottom, there are two buttons: a blue 'Register' button on the left and a blue 'Forgot Password' button on the right.

Figure 3: User Login Page

During registration we had implemented multiple checks where the database schema and validations in place wouldn't allow two users having the same username and the password to be less than 9 characters and no backslashes as well which prevents from SQL injection attack. Multiple registrations with same details are also prevented.



The image shows the 'Initial Setup' page for 'Latex-Made-Easy'. At the top, there's a header with the text 'Latex-Made-Easy'. Below the header, there are two buttons: a blue 'Save Later' button and a blue 'Get Later' button. Below these buttons, there are four input fields: 'Title:', 'Author:', 'College:', and 'Degree:'. Below these fields, there's a 'Date:' field with a calendar icon and the date '3/9/2018'. At the bottom, there are three buttons: a blue 'Go Previous' button, a red 'Log Out' button, and a blue 'Next >>>' button.

Figure 4: Initial Setup

After logging successfully, the user would be provided with all the template components in an orderly way. The user can write the content in the text boxes provided. At each step in finishing the paper, the user has an option to leave some sections. Also, the user can download the latex code for each section if wanted. This design plan would therefore help people to get Latex codes for simple components such as tables, lists etc.

Figure 5: Basic Elements

Now once the user navigates through the starting pages where they have optional elements to enter according to their need like dedication, acknowledgement, degree and university. The major content formatting is supported in a single page where the user can add an introduction and number of chapters and under each chapter the option to add multiple sections and tables and lists wherever needed. This page covers the major part of the paper or report being prepared which is also shown in the figure below.

Save LaTeX

Get LaTeX

Add Chapter

New Chapter 1:

Hide Chapter

Add Section

Add Table

Add Paragraph

Add List

Remove Chapter ✖

Introduction:

Show Introduction

Remove Introduction ✖

New Section 1:

Hide Section

Remove Section ✖

<<< Previous

Next >>>

Log Out

Figure 6: Main Section Formatting

During research works and project reports, analysis of various data is mostly needed, this adversely creates the need for tabulating the data which needs a table in the report. So since this is one of the primary need for reports we have embedded this feature, where upon adding a table will further enable options to add columns and as well as to add rows according to the user's need. Further these views can be minimized so that the user has complete view which ensures

user knows where each of the elements are present. This further helps the user to not miss out any elements needed for his/her report.

New Table

Hide Table

Add Column

Add Row

Remove Caption ✖

Remove Table ✖

Caption:

Caption Please

Column 1	Column 2	Column 3	Column 4
Double Click to Edit	Double Click to Edit	Double Click to Edit	Double Click to Edit
Double Click to Edit	Double Click to Edit	Double Click to Edit	Double Click to Edit

<<< Previous

Log Out

Next >>>

Figure 7: Table Entry

One other feature we realized that users would be primarily using is the lists, while there is different forms to represent the list which is ordered or unordered list. Users will have the option to select which kind of lists would they like to use. Based on their input the latex code is developed accordingly. This feature is majorly useful for users formatting their research documents since users creating project reports are less likely to use lists because there is less research involved in project reports. So this factor made us embed this particular feature along with the existing features.

List 1:

Hide List

Add Item

Remove Last Item ✕

Remove List ✕

☒ Unordered ☐ Ordered

Item: 1

<<< Previous

Next >>>

Log Out

Figure 8: List Entry

One other most important feature for user would be an option to save his/her work and given the depth and work

involved in any research or a project, writing a report in a stretch in not a feasible option so any user would like to save the work and come back to continue. This is provided right beside Get latex where users can save their work. For preventing data being lost, entered data is also automatically saved under the database.

Now after providing the content to all the sections prompted at each stage, the user is provided a way to download the latex code in a zip file with an option as Get Latex. This zip file will contain the latex code of the elements needed which can be tested by copying the code to any latex editors and compile.

3. USER EVALUATION

Under this section we will be discussing what were the elements we wanted to evaluate on and the user evaluation plan. Since the application was build with various factors being assumed and considered, gathering user experience data and feedback is very crucial and also the target participants of this evaluation has also been discussed.

3.1 Evaluation Plan

To test our web application, the multiple factors which encouraged us in the first place to carry out this project will be the critical elements based on which we would be evaluating the final output. So since the application presently deals with only one template the target participants will also be students who have had experience using latex. We are firstly interested in knowing whether users would actually like to have a application for formatting their documents. Along with this one of the most important concern of ours is to evaluate whether time could actually be saved when users use this application instead of traditional Latex editors.

With the initial interaction of user with the application we are concerned whether the login process and registration meets the expectations and requirements of the users. Since a smooth flow of this process actually impacts the whole user experience. So we were interested in this particular section too though it is a little off from the actually purpose of the application, but it does adversely affect on the experience on the whole.

As the user logs in, to make it as simple as possible and at the same time making sure the user can easily understand the interface the design is made with simple prompts to next pages in a orderly fashion, but this approach might be or may not be easy to adapt. So in order to make sure the interface was actually easy to adapt we wanted users to evaluate on this particular section too. Since the main purpose of the application is to make the process of formatting documents and easier and faster.

That being evaluated, the complexity and the scope of Latex is large and hence as we focused on certain features assuming these would be of primary use for majority of the users. We definitely wanted to know whether our assumptions regarding the same was actually true by evaluating the users whether the features implemented right now actually meet most of their requirements.

After anyone creates an application, it should actually have the capability to attract users to use it compared to other existing systems. So now given that users would ac-

tually like to have an application based on initial project idea survey. We were interested to know whether they would prefer to use this application from the next time they need to format their documents using latex.

Finally since we know that these features might not meet the user requirements completely and there might be still a lot other features that users would like to have, we wanted to know what other major features would users like to have so that they further don't have to go to any other latex editors at all for their complete need. Following with this objective, users were welcomed to give any other suggestions which will enable this project to further being enhanced so that it becomes a complete product with all the features in the future.

3.2 Participants

The majority of the participants in the user evaluation are software engineering students in Spring Semester 2018. There are graduate students doing computer science at North Carolina State University. Hence most of them are well versed with using wide range of web applications and especially since they would have had first hand experience with Latex at some point of time the evaluation would be completely reliable based on the data gathered from these participants.

Additionally the participants were also from various technical backgrounds since the experience with latex varies from a student with computer science background and someone with economics background since it might be little easy for users with computer science background to understand the latex coding syntax when compared to someone who is not form computers background. So the data gathered will cover various perceptions.

4. RESULTS

Based on the evaluation plan we have decided to survey those questions through a google form and collect the data required. Users when asked whether they thought the application would be useful for someone writing a technical paper, 53.8% of users rated 5 on 5 while 46.2% of users rated it 4 on 5 with respect to this question as shown below.

On a scale of 5, How useful do you think this application would be to someone writing a technical research paper?

26 responses

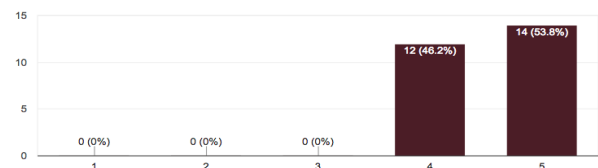


Figure 9: Productivity of the application

When asked about how likely the users think this application might save their time. There were a broader range of responses from ratings starting at 7 till 10. Majority users that is 34.6% of users rated 8, 26.9% of users rated 9 and

30.8% of users rated 10 on 10. While a very few users which is 2.2% have rated it a 7. The distribution can be seen in the following diagram.

How likely do you think using this application would save your time in formatting your research paper?

26 responses

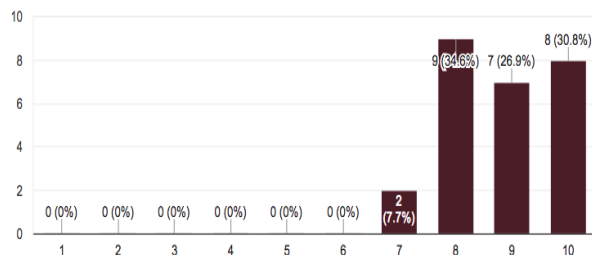


Figure 10: Amount of time saved

Regarding the login portal and registration process 88.5% of users have stated that they were easily able to register and login. While a few users which comprise of 11.5% of users found it easy but thought it could be made more comfortable.

Was it simple for you to register and login or did you face any difficulties?

26 responses

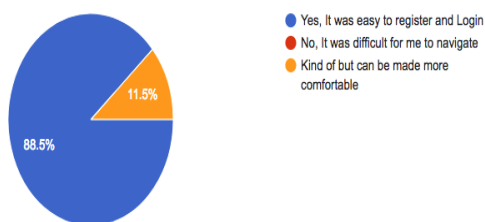


Figure 11: Login Issue

The whole User Interface was easily understood and adapted by majority of users comprising of 69.2% of the users while 30.8% though it could be made better so that it becomes much easier to adapt and use.

Were you able to easily understand and adapt the user interface after logging in?

26 responses

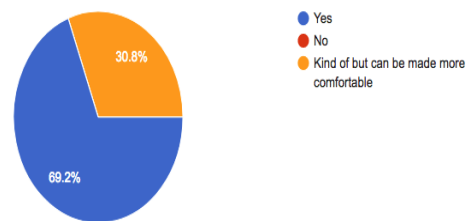


Figure 12: UI Adaptability

Regarding the features presently implemented, 61.5% of the users felt that they could find majority of the features needed by them while 38.5% found most of the features but not everything they needed.

Were the features implemented right now meet your major requirements?

26 responses

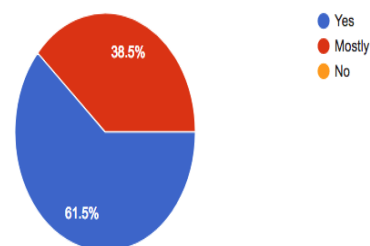


Figure 13: Feature Sufficiency

When asked whether users would prefer using this application for formatting their documents, 61.5% of users positively responded stating they would prefer using it. While 34.6% of users stated that they would most probably use the application and only 3.8% of users were speculative and stated they may consider using the application.

Would you consider using this application for formatting your reports?

26 responses

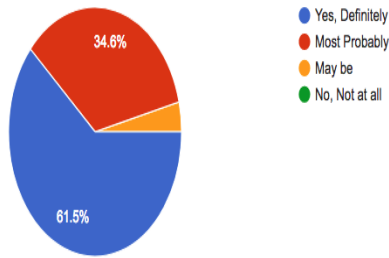


Figure 14: Tendency to use the application

The users on pointing out the some of the impressive features in the application came up with some of the features they thought were worth and probably the reason why this application would be a better alternative to the existing latex editors.

Are there any sections which you thought were really impressive?

21 responses

The main impressive feature of the product was that we can get the latex file in zip file which can be uploaded on the shareLatex file.

Adding modules

The real time heading and section creation. Reduces significant time in doing redundant commands for creating tables and sections.

The application is developed in such a way that it generates error free code

faster way to get latex code for a specific component in a thesis paper

Hide chapters button in the Chapters page was impressive

Adding multiple chapters is very useful for IEEE members

Hiding chapters button is very useful, Most of the section in latex have been implemented except math

I found the entire concept and idea of this project impressive. Very nicely done.

Figure 15: Impressive Features

Users upon asking to suggest which features they would like to be seen embedded further in the application. Users had varied choice of responses but majority of the users wanted to have mathematical equations and symbols to be included.

What elements do you think should definitely be added so that Latex-Made-Easy becomes everyone's 1st Choice?

26 responses

It has only one template. Adding more templates would be very useful.

Ability to download the pdf as well.

Drag and drop of elements as well as equations which would make it really useful.

You can make a word extension which can be added to make the latex code of the word file currently editing.

Remove elements, PDF download

Supporting incorporation of images

This application will reduce the time to create latex docs.

add symbols and way to add expressions

Saving multiple documents in a single session

Mathematical equations

Figure 16: Features Desired

5. DISCUSSION

5.1 Observations and Limitations

Based on the reviews given by the users after using the application, it gave us a lot of insight with much better accuracy. When asked whether they wanted to have an application, a majority portion of the students showed interest towards it. Additionally from the ratings received regarding the amount of time being saved by them by using this app, we could conclude the overall rating to be more than 8.75 out of 10 in terms of time saved for users. That adversely states that majority of the users do think this application will actually save their time.

UI is major part for any application because a non-interactive UI sometimes causes a user to assume that the application is not useful. From the survey results we feel that we have accomplished this part quite impressively. The survey states that above 83% of participants felt UI was very intuitive. We included many special features in chapters session like to hide a subsection of a chapter which we think a majority of users liked it. This being said, we got a good amount of suggestions for UI like adding drag and drop feature. This feature may help us to implement a major task in future scope that helps in incorporating many other templates.

Also in our analysis, we included students from different branches who are doing advanced level courses since they would majorly demand Latex to present their final work. All the reviews we received from them are positive. All three of them reported that it substantially saved their time to write the latex documents and also the user interface was intuitive. Having an intuitive interface will be very advantageous as the application grows. Also the major feedback we received from these users is offering an option to save multiple documents in a single section which we already included in our future scope section.

5.2 Best Feature

Majority of the users felt skipping the section if not required as the best feature implemented. It strengthens one of our core reasons to start our application which was to generate Latex code for individual components in latex for the user who is not well versed with latex or not having any prior Latex editing experience before. With this feature user can get only code for specific section.

Other impressive features found by users were the ability to hide elements and also the real time creation of section and chapters since that reduces their overhead time of repetitive code addition of such elements.

6. FUTURE SCOPE

People from various technical backgrounds use latex to write different formats of data. This project at present only deals with one such format while latex can be possibly imagined to format any document you need. Providing drag and drop facility in addition to current implementation might help to generalize the application to a lot of templates.

Also allowing users to save multiple documents in a session would make the life simpler because many users may prefer to write many version of a file initial to have many test versions.

Converting latex code to different document formats like pdf, word docs would make many users waste less time because they need to navigate to other site to do conversions. Also, this feature individually can benefit a lot because this is one of the major requirements for many users.

One other major future scope will be to provide an interface for users to be able to view their latex documents in the same application in real time.

Mathematical equations are something that is unique to latex in terms of formatting. Although we tried our best, we missed to implement it due to the time constraints. This feature is very useful to a lot of scholars and would redefine the way this application is used.

7. CONCLUSION

The major focus of the application will be to provide a portal to the users where they can add the components to a template similar to the graduate-thesis template of Sharelatex. Using this application, the users should be conveniently able to create the graduate thesis without worrying about using Latex. The users should also be able to generate pdf for the thesis paper. As the application will support a large number of functionalities, the users should be able to find a way to add any type of content they want to be added to the paper. Given the advantage over traditional latex editors, though the present system doesn't support all the functionalities yet but it has a lot other advantages, when compared to present existing traditional editors as shown below.

Application	Convenience	Time	Sharing	Security
ShareLatex	+	+	+	+
OverLeaf	+	+	+	-
TexMaker	+	+	-	-
Latex-Made-Easy	++	++	-	+

Table 1: Comparison

Also the application is primarily designed to make sure people can just have a specific component in a thesis paper and provide the content for it and are able to get the latex code. This would remove the burden of searching for a template that have the component and again going through the complete template i.e to look through the codes and then copy the latex code required.

Though there are lot of online Latex editors available with multiple sites providing only certain elements like equations or tables. This application with further development can potentially be a game changer in this particular field. We look forward to see this application being developed and probably replace the traditional Latex editors in the market one day.

8. REFERENCES

- [1] Latex
<https://www.latex-project.org/get/>
- [2] Ben Trovato, G.K.M. Tobin, Lars Thörvald, Lawrence P. Leipuner, Sean Fogarty, Charles Palmer *Alternate ACM SIG Proceedings Paper in LaTeX Format*
- [3] M. Trettin (2007) *An essential guide to L^AT_EX2_ε usage - Obsolete commands and packages.*
- [4] N. Talbot (2004) *Creating a PDF document using PDFLATEX*
- [5] Lapo F. Mori *Writing a thesis with L^AT_EX*
- [6] <https://en.wikipedia.org/wiki/LaTeX>
- [7] <https://en.wikipedia.org/wiki/TeX>
- [8] L. F. Mori (2007). *Tables in LaTeX2_ε: packages and methods. The PracTEX Journal,*
- [9] <https://tex.stackexchange.com>
- [10] <https://yelkhatib.wordpress.com/2017/01/21/collaborative-latex-editors/>
- [11] Handlebars
<https://handlebarsjs.com/>
<https://javascriptplayground.com/javascript-templating-handlebars-tutorial/>
- [12] Travis CI
<https://travis-ci.org/>
- [13] Coveralls.io
<http://coveralls.io/>

9. CHITS

1. HRQ
2. HIO
3. RNJ
4. LGS
5. OVV
6. ICB

7. FGE
8. GOX
9. MVZ
10. BZD
11. PAB
12. VQC
13. LRS
14. GZJ
15. SMC
16. VPR
17. CZP
18. KUM
19. XCO
20. ZDI
21. URP
22. NWC
23. CWW