**Feasibility Study**

**Team Shiners**

**MSUClassIC**

By Athul Sajikumar, Marcos Lopez, Samuel Olatunde

TABLE OF CONTENTS

**1. Introduction**

1.1. Overview of the project

1.2 Objectives of the project

1.3 The need for the project

1.4 Overview of existing systems and technologies

1.5 Scope of the project

1.6 Deliverables

**2. Feasibility Study**

2.1 Financial Feasibility

2.2 Technical Feasibility

2.3 Resource and Time Feasibility

2.4 Risk Feasibility

2.5 Social/ Legal Feasibility

**3. Considerations**

**4. References**

1. **Introduction**

1.1. Overview of the project

MSUClassiC will be a scheduling app for MSUTexas designed to simplify scheduling for Registrars, College Deans, and Department Chairs at Midwestern State University. Users need to schedule every semester at MSUTexas, requiring them to allocate faculty, classes, and rooms at specific times. This process is currently tedious and frustrating for many users. Our app aims to streamline this process by providing a user-friendly interface and customizable options for scheduling and modifications.

1.2 Objectives of the project

The objective of this project is to:

* Focus on all three user groups (Registrars, College Deans, and Department Chairs) to improve workflow for each group.
* Create different views tailored to the needs of various users to ensure ease of use and usefulness.
* Design a non-technical user interface to enhance accessibility.

1.3 The need for the project

Academic Scheduling is a complex process. The most convenient and effective way to handle class scheduling is through online applications. However, many similar apps are overloaded with features like optimizing data analytics and event planning, making them complex and difficult to use. To address these issues, MSUclassIC prioritizes class handling and scheduling. We focus solely on providing solutions for Registrars, Deans, and Chairs, offering them cleaner, non-technical user interfaces with enhanced data visualization capabilities.

1.4 Overview of existing systems and technologies

Existing systems are overloaded with features like optimizing data analytics and event planning, making them complex and difficult to use. The current solutions in use by many universities all contain the same complexities that hinder their wide adoption by their intended users. These applications: AdAstra, Coursedog, Accruent EMS, and Course Leaf are all intended for a wider user-base and, as a result, involve very complicated structures and poor user interfaces.

Each of these competitors are being sold to the upper administration as a solution to all scheduling within the university. This leads to an implementation that must work for every potential scheduler on the campus. Since the requirements are so much more, then the software must be more complex. Some products, like AdAstra and Accruent EMS have poor user design making their web application harder for users to use.

1.5 Scope of the project

The main actors of this system are users, with key use cases including login/logout, personalized views, request submission, conflict viewing, and request acceptance/denial. The deliverable is a web-based application with a user-friendly interface.

Main actors of this system – User’s

Main use cases associated

User’s can

* Login and logout
* View personalized views depending on the user's role
* Submit requests to the registrar regarding class changes, as the user is also a registrar
* View schedule conflicts, if any
* Accept or deny requests based on conflicts

1.6 Deliverables

A web-based application. This system consists of various functionalities. Very easy and non-technical UI will be provided to users.

**2. Feasibility Study**

2.1 Financial Feasibility

Our course scheduling system will be available for free, with the option to pay for extra features later on. While there will be some costs for fixing bugs and maintaining the system, we plan to start by offering our service to schools and colleges. Even though there are some costs involved, our customers will gain a lot from using our system, like easier scheduling and better organization. So, overall, our project is financially feasible

2.2 Technical Feasibility

Project MSUClassIC is a complete web-based application. The main technologies and tools that are associated with the MSUClassIC are

* + -  HTML
  + -  PYTHON
  + -  Vue.js
  + -  MYSQL
  + -  Diagram Drawing tools
  + -  Draw.io

Each of the technologies is freely available and the technical skills required are manageable. Time limitations of the product development and the ease of implementing using these technologies are synchronized.

From these, it’s clear that the project MSUClassIC is technically feasible.

2.3 Resource and Time Feasibility

Resource Feasibility  
Resources that are required for the Calorie tracker project include,

* -  Programming device (Laptop)
* -  Hosting Space (freely available)
* -  Programming tools (freely available)
* -  Programming individuals.

So, the project MSUClassIC has the required resource feasibility.

As per the business request, it needs to be done by May 9th.

2.4 Risk Feasibility

1. Risk feasibility can be discussed in several contexts. The risk associated with the size.

Estimated size of the product in the line of codes:

MSUClassIC is a web application with many numbers of stakeholders, MSUClassIC will contain a significant amount of code lines. As the system doesn’t contain any multimedia aspect, the file sizes and the complete project size will not exceed 200MB

Estimated size of product in a number of programs:  
Though the application supports many stakeholders, it will be constructed as a single web application with a single login page rather than having many numbers of sites for the users.

The Database size will not exceed values that are supported by Vue.js, and the number of relations and entities are minimized by using best practices of normalization theories.

1. Users of the product:

-  Registrar’s.

-   Deans.

-   Chairs.

-   Administrators.

1. Number of projected changes to the requirements for the product? Before delivery? After delivery:

The requirements are identified before the implementation phase. Being a general product (not specific to a single user), the requirements will be changed only if new functionalities are added to the system.

1. Amount of reused software:

Though the main logic’s are implemented throughout the project, MSUClassIC, Will use some Vue.js libraries (Crispy forms) to incorporate additional functionalities such as to support the file uploads, and the data implementation needed by the user (signup, login)

1. Business impact risks  
   1. Effect of this product on company revenue:

MSUClassIC can be implemented as an individual system or can be integrated to an existing system. Since it automates some key features in the Academic process.

* 1. Reasonableness of delivery deadlines:

Being a 14-week project, the project MSUClassIC will have several deadlines and deliverables that are scheduled successively. Depending on the coding and designing cost and effort, the deadlines are quite reasonable.

* 1. The number of customers who will use this product and the consistency of their needs relative to the product:

* 1. As mentioned above, we can categorize stakeholders into 3 main categories. This system can support many numbers of users simultaneously due to low bandwidth requirements.
  2. The number of other products/systems with which this product must be interoperable: This product can be seamlessly integrated with existing apps with minor modifications This integration will enhance the user experience for both systems, adding significant value to end users.
  3. The sophistication of end users:

MSUClassIC is designed while maintaining the complexity at a very low level. Usability is highly improved by providing help documents and making GUIs easy to use.

* 1. Amount and quality of product documentation that must be produced and delivered to the customer:

Customers will be provided with a complete online user help page. As the software is implemented as a freeware and open-source system, the code will be available for free.

1. Costs associated with delivery:

At the initial stage, there would not be a cost associated with the app, the only cost will be the hosting cost.

1. Customer-related risks:

MSUClassIC is a general-purpose product, not specifically tailored for one customer. While we are currently developing it to align with MSU's workflow, implementing the system in other educational institutes may require basic modifications to adapt to their unique environments.

1. Development environment risks:
2. Is a software project management tool available?

Notion will be used as the main project management tool.

1. Are tools for analysis and design available?

MSUClassIC will require several designing software:

* + - * Draw.IO (database design and class diagram
      * Photoshop (Software related diagrams)

1. Are compilers or code generators available and appropriate for the product to be built?

Python will be used as the main scripting language. All the libraries and I Interpreters will be freely available.

1. Are the testing tools available and appropriate for the product to be built?

Django is the main testing tool that will be used. Django is a freely available tool that supports automated testing.

1. Are software configuration management tools available?  
   Configuration management will be done using HTML that is freely available.

1. Does the environment make use of a database or repository?

This is a database-oriented system that will use MYSQL

1. Are all the software tools integrated with one another?

Main deliverable will be packaged under a single project. All the stakeholders will have a single login page.

1. Process issue risks

MSUClassIC will follow the software development process. This provides the flexibility to accommodate changing software requirements of MSUClassIC

1. Technical issue risks
2. Are specific conventions for code documentation defined and used?

Software code will be freely available, and the code documentation will be provided.

1. Do you use a specific method for test case design?

Visual Studio will be used as the main testing tool that automates the testing process.

1. Are configuration management software tools used to control and track change activity throughout the software process?

Python, HTML, Firebase and JavaScript will be used throughout the software implementation process.

1. Technology risks
2. Is the technology to be built new?

All the technologies are very well established.

1. Do the system requirements demand the creation of new algorithms, input, or output technology?

MSUClassIC will have several algorithms to facilitate a request system and to manage a request queue.

2.5 Social/ Legal Feasibility

MSUClassIC uses freely available development tools and provides the system as an open-source system. Visual Studio libraries that are used in this system are free open-source libraries, which will have a great impact on a academic system.

**3. Considerations**

3.1 Performance:

MSUClassIC requires a very low bandwidth; hence the performance will not degrade with an increasing number of potential users. We can use firebase to host the website currently, it's free for small applications (which we are), and if MSU were to adopt this software, they would host it on their servers, and we would migrate both hosting and authentication to the MSU portal

Response time: less than 2 seconds

Processing time: Less than 2 seconds (no bath processing involved) Query and reporting times: yet to be tested.

Throughput: yet to be tested

Storage: yet to be tested.

3.2 Security:

Security measures are provided in many aspects of this system.

User authentication:

Users will have to authenticate using the username and password. Depending on the access level each user will gain the functionality of the system.

Login details:

Each user’s login time and logout time will be recorded in the system, to make the tractability process easy in case of a faulty action.

3.3 Usability and ease to use:

The interfaces are designed to make it easy for any potential user to get familiar with the system within 5 mins.

3.4 Capacity and scalability:

MSUClassIC can accommodate many simultaneous users. The system is designed to make it easy to integrate into an existing system

3.5 Availability

The system will be available for 24 hours. Mean time to failure and mean time to repair will be decided to increase the availability. With a paid hosting space, the availability can be guaranteed to be a great passion.

3.6 Maintainability:

MSUClassIC is designed using the best practices of Python and HTML. Since every single segment in the system is very well structured, the system is highly maintainable.

**4. References.**

<https://www.slideshare.net/PasinduTennage/sample-software-engineering-feasibility-study-report>