

# **ALLTHINK DOCUMENTATION**

ALLTHINK DOCUMENT .....	1
I. OVER VIEW: .....	4
1. Mission and Scope .....	4
2. Development method .....	4
3. Status .....	4
4. Resources and schedule .....	4
5. Design .....	4
6. Quality Assurance .....	4
7. User Support .....	4
II. DEMO SCRIPT .....	5
1. Release Information.....	5
2. Demo Planning.....	5
3. Anticipated Questions.....	6
4. Demo Checklist .....	6
III. DESIGN.....	6
1. Introduction .....	6
2. UML Structural Design .....	6
3. UML Behavioral Design.....	8
IV. DESIGN ARCHITECTURE.....	9
1. Component .....	9
2. Deployment.....	9
3. Integration .....	9
V. DESIGN SCALABILITY.....	10
1. Relevant parameters .....	10
<b>Parameter</b> .....	10
<b>Description</b> .....	10
Registered users .....	10
Depend on host .....	10
Registered users .....	10
Depend on host .....	10
Map size .....	10
Depend on host .....	10
Game pieces .....	10
Depend on host .....	10
2. Scalability Mechanisms.....	10
VI. DESIGN SECURITY .....	10
1. Physical security mechanisms .....	10
2. Network security mechanisms .....	11
3. Application security mechanisms.....	11
VII. DESIGN UI .....	11
VIII. USER FAQ .....	13
1. General Information .....	13
2. Download and Install.....	14
3. Getting Started .....	14
4. Troubleshooting .....	14
5. Other Questions .....	14
IX. IMPLEMENTATION NOTES .....	15
X. RELEASE NOTES .....	15
1. Introduction .....	15
2. What's New?.....	15
3. Installation and Upgrade Notes.....	15
4. Recent Changes.....	16
XI. TEST CASE.....	16

1.	<i>Login-1: Normal User Login .....</i>	<i>16</i>
2.	<i>Login-2: Locked-out User Login.....</i>	<i>17</i>
3.	<i>Viewing the Lesson .....</i>	<i>17</i>
4.	<i>Accept the registration and add new lesson.....</i>	<i>17</i>
5.	<i>Changing avatar .....</i>	<i>18</i>

# I. Over view:

## 1. Mission and Scope

- Project goal: Anyone can make their own lessons and share it for lots of people.
- Project scope: Anyone can use these applications!

## 2. Development method

- Agile method: Extreme programming.

## 3. Status

- We are trying to complete this project best.
- Status reports:
  - Our first report in the 6th week of this semester.
  - Our second report in the 10th week of this semester
  - Our last report in the 15st week of this semester.

## 4. Resources and schedule

- The deadlines for this project: 15st week of this semester.
- Members:
  - ❖ Nguyễn Anh Tuấn.
  - ❖ Nguyễn Thế Tùng.
  - ❖ Vũ Tiến Tùng.
  - ❖ Trịnh Văn Tú.

## 5. Design

- ✓ Using diagrams document:
  - ❖ *UML class diagram*
  - ❖ *UML class diagram*
  - ❖ *UML state diagram*
  - ❖ *UML sequence diagram*
  - ❖ *UML deployment diagram*
  - ❖ *Other design*

## 6. Quality Assurance

- About test cases: We will use **selenium HQ** to test and demo in the least week.

## 7. User Support

- User documentation:
  - ❖ *Anyone can get it from :* [https://github.com/tuanna55/-k55-INT2026\\_N9\\_Allthink](https://github.com/tuanna55/-k55-INT2026_N9_Allthink)
  - ❖ Or access <http://allthink-g9.herokuapp.com> to use this application.
- Technical support or report problems:
  - ❖ Support: contact us.
  - ❖ Issue tracking: contact us.

## II. Demo Script

### 1. Release Information

Project:	<b>ALLTHINK</b>
Internal Release Number:	version 2.1
Related Documents:	visit here get more information:  <a href="https://github.com/tuanna55/-k55-INT2026_N9_Allthink">https://github.com/tuanna55/-k55-INT2026_N9_Allthink</a>  or <a href="http://allthink-g9.herokuapp.com">http://allthink-g9.herokuapp.com</a>

### 2. Demo Planning

- Our teacher and our classmates is the target audience for this demo.
- Our main goal of demo planning:
  - ❖ To convince the teacher that we are trying our best to do this project.
  - ❖ To inform the teacher and our classmates about the project and prompt further discussions. E.g., showing another team where their product would have to integrate with ours, showing how we work together...
  - ❖ To provide evidence of progress. E.g., showing our teacher that you are making effort to do this project.
- To achieve our goals:
  - ❖ This project automates much of the user's task, thus increasing productivity and cutting costs.
  - ❖ This project is very easy to use, which will encourage rapid adoption.
  - ❖ We have built 7 out of 10 features, and it turns out that one of the remaining features cannot possibly work like the others.
- Demo time: 15 minutes, have questions

- Demo process: All group member, a computer with a web browser.

### 3. Anticipated Questions

- Comparing with other project: We will compare it with another project of our classmates.
- Finishing: We have announced a release date at the end of this semester.

### 4. Demo Checklist

- ✓ *Are you making the right points in the right order?*
  - Yes, we feel that we will accomplish our goals
- ✓ *Can it be presented in the available time?*
  - Yes, we could even add more or finish early.
- ✓ *Do all system features actually work as needed?*
  - Yes, everything is finished and working.
- ✓ *Does the demo end effectively?*
  - Yes, viewers will be satisfied and convinced.
- ✓ *Does the demo raise questions that it does not answer?*
  - Yes, the goal of this demo is to raise questions and issues that we will then work to resolve.

## III. Design

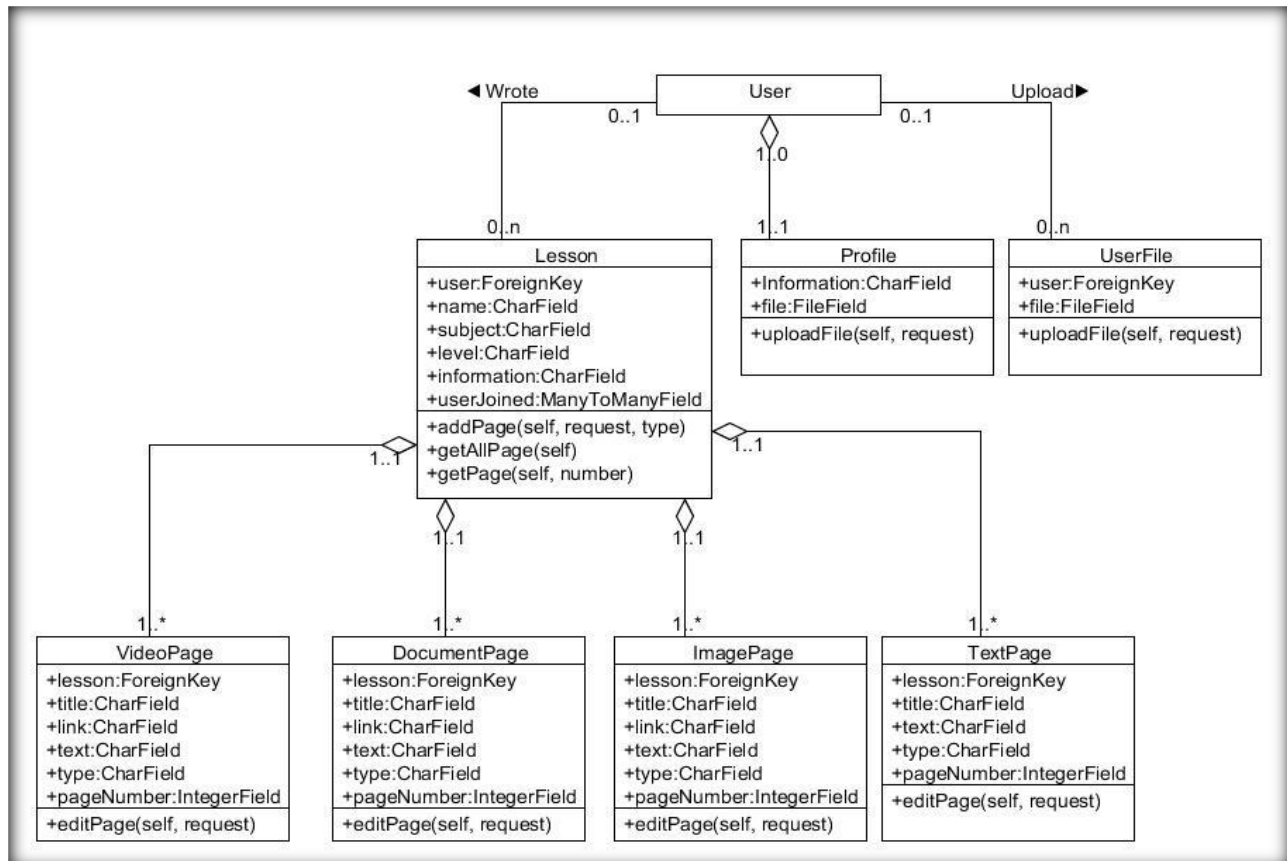
### 1. Introduction

- This main page describes the system design in terms of packages, classes, relationships, and behavior. Several attached worksheets address specific aspects of the overall system design, such as user interface and database design.
- The prioritized goals of this design:
  - ❖ [Correctness](#)
  - ❖ [Feasibility](#)
  - ❖ [Understandability](#)
  - ❖ [Implementation phase guidance](#)
  - ❖ [Modularity](#)
  - ❖ [Extensibility](#)
  - ❖ [Testability](#)
  - ❖ [Efficiency](#)

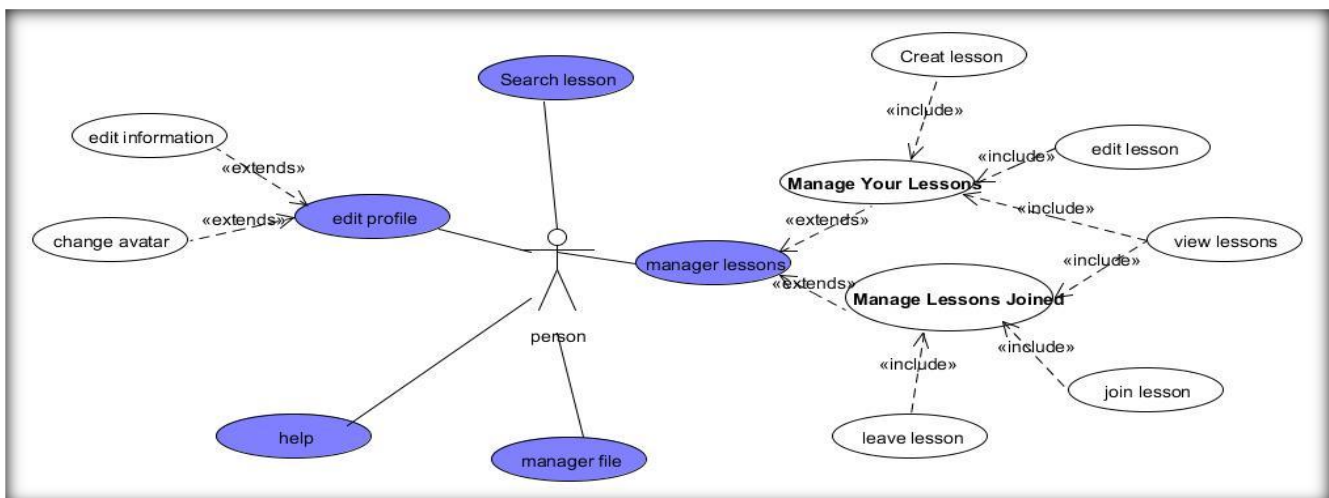
### 2. UML Structural Design

The system's Structural design is described in the following UML Structural diagrams:

### *Class Diagram*



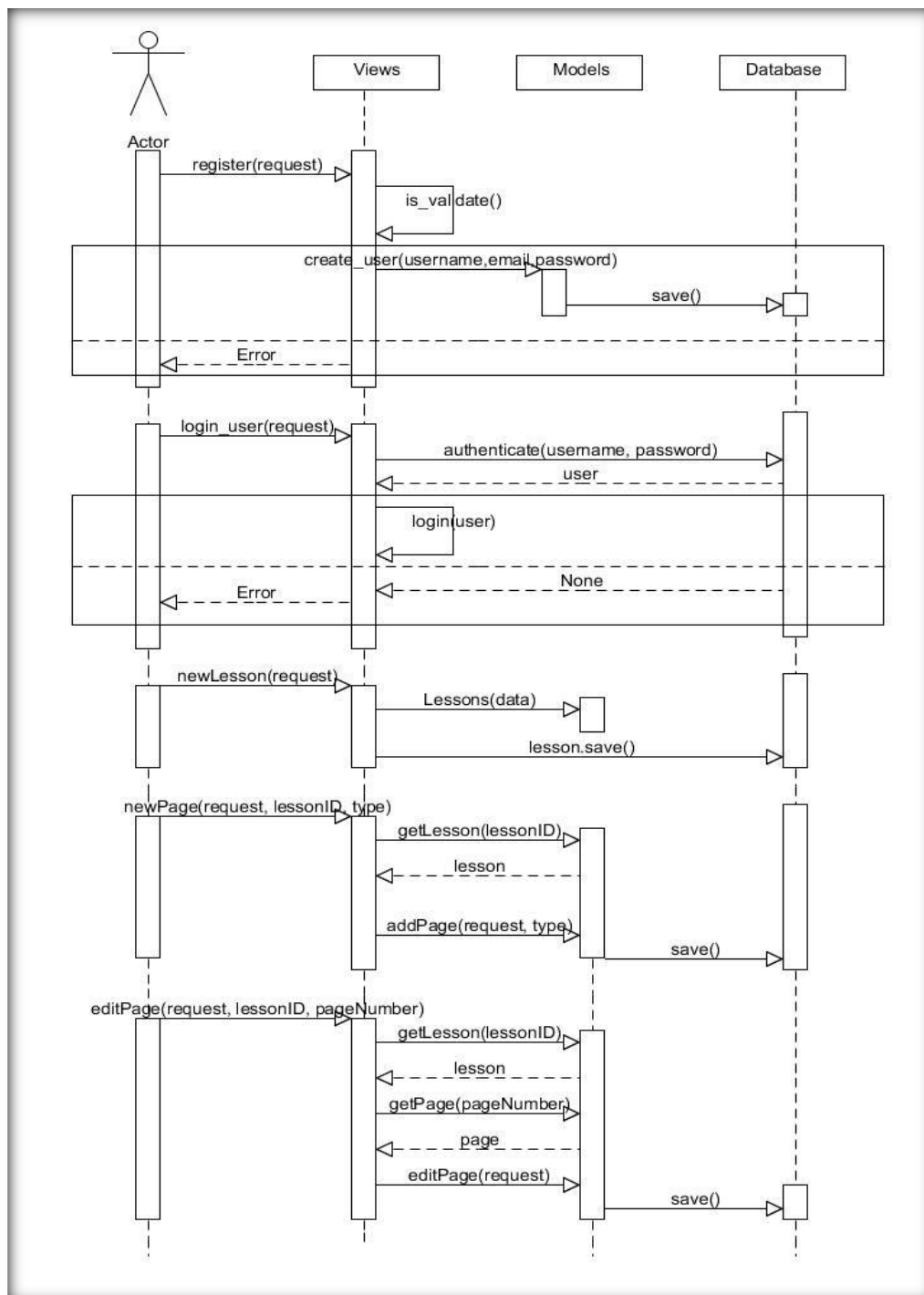
### Usecase Diagram



### 3. UML Behavioral Design

The system's behavioral design is described in the following UML behavioral diagrams:

#### *Sequence Diagram*



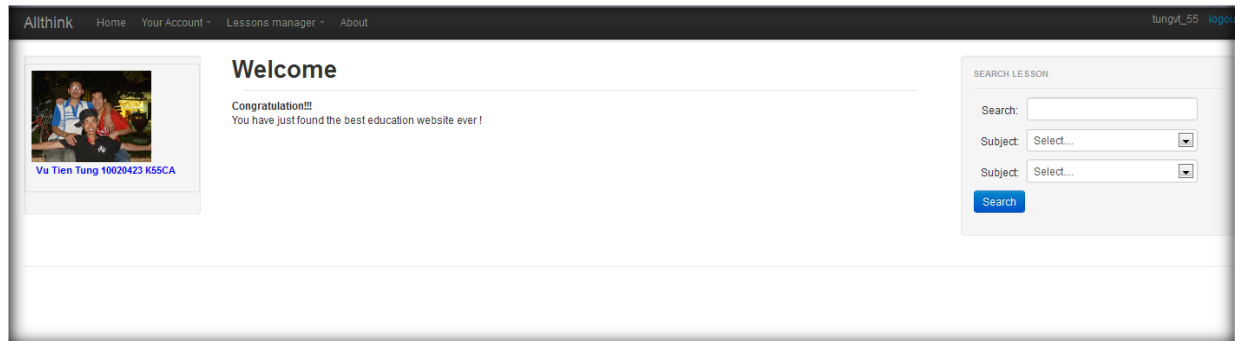


## IV. Design Architecture

The architecture we build for this application by on client-server architecture.

### 1. Component

- User Interface Component includes modules as : avatar, account, lesson and search.



### 2. Deployment

- *What aspects/resources of their environment are shared?*
  - ❖ Everything is on one server so all machine resources are shared by all components.
  - ❖ All machines share the same bandwidth to the Internet. All machines access the same file server. So, if one component uses the resources heavily, other components may have to wait.
- *How are requests allocated to redundant or load-balanced servers?*
  - ❖ We are not doing any load-balancing or redundancy for fail-over.
  - ❖ Load-balancing among front-end servers is handled by a load balancing device that we can make very few assumptions about. However, once a user session is established, the same front-end server will be used for all requests during that session.
- *What alternative deployment configurations are possible?*
  - ❖ This is the only possible deployment.
  - ❖ The database could be moved to a different machine with a fairly simple change to a configuration file. Otherwise, nothing can be changed about the deployment.
  - ❖ We have the ability to move the database process to a separate machine. We have the ability to add more front-end servers. The application logic running on the application server cannot be split or load-balanced.

### 3. Integration

- *How will components be integrated? Specifically, how will they communicate?*

- ❖ All of our code uses direct procedure calls. The database is accessed through a driver.
- ❖ Components within the same process use direct procedure call or standard Java events. Plug-ins are also accessed through a API of direct procedure calls and standard events. Communication with the database uses a JDBC driver. Communication between the front end-and back-end servers uses XML-RPC.
- *What architectural mechanisms are being used to ease future extensions or modifications?*
  - ❖ We could change the database by switching drivers. Otherwise, extensions and modifications can only be done at the design level.
  - ❖ New front-end components could be added so long as they access the back-end the same way. New plug-in components can be dynamically loaded, so long as they satisfy the plug-in API. Otherwise, there is no ability to add or exchange components, because this architecture uses direct dependencies between its components rather than implicit invocation. Extensions and modifications can be made at the design-level, but deploying those changes requires recompilation and down-time.

## V. Design Scalability

### 1. Relevant parameters

<i>Parameter</i>	<i>Description</i>
Registered users	Depend on host
Registered users	Depend on host
Map size	Depend on host
Game pieces	Depend on host

### 2. Scalability Mechanisms

Action	Goal	Time Formula	Description
Login	1 second	1 second	½ second
Display map	1/5 second	1 second	1 second

## VI. Design Security

### 1. Physical security mechanisms

- Servers will be kept in a locked room with door code known only to administrators.
- Servers will be kept in a locked equipment rack.
- Server case itself has a security cable that prevents the case from being opened (so the hard-disk with sensitive data cannot be removed).
- Backup tapes are stored in a locked cabinet in a locked room.

## 2. Network security mechanisms

- A firewall device limits access to specific network ports (e.g., port 80 for web server access).
- Firewall software limits access to specific network ports (e.g., port 80 for web server access).
- Only the front-end machines are accessible over the Internet. Other machines in the server cluster communicate over a private LAN only.
- Users can only connect to the server from specific ranges of IP-address (e.g., university-owned computers in networks on campus).
- Certain users (e.g., administrators) can only connect from specific ranges of IP-addresses.
- All network communication takes place over a virtual private network (VPN) that is encrypted and not accessible to outsiders.
- All network communication takes place over a LAN that does not have any connections to the Internet

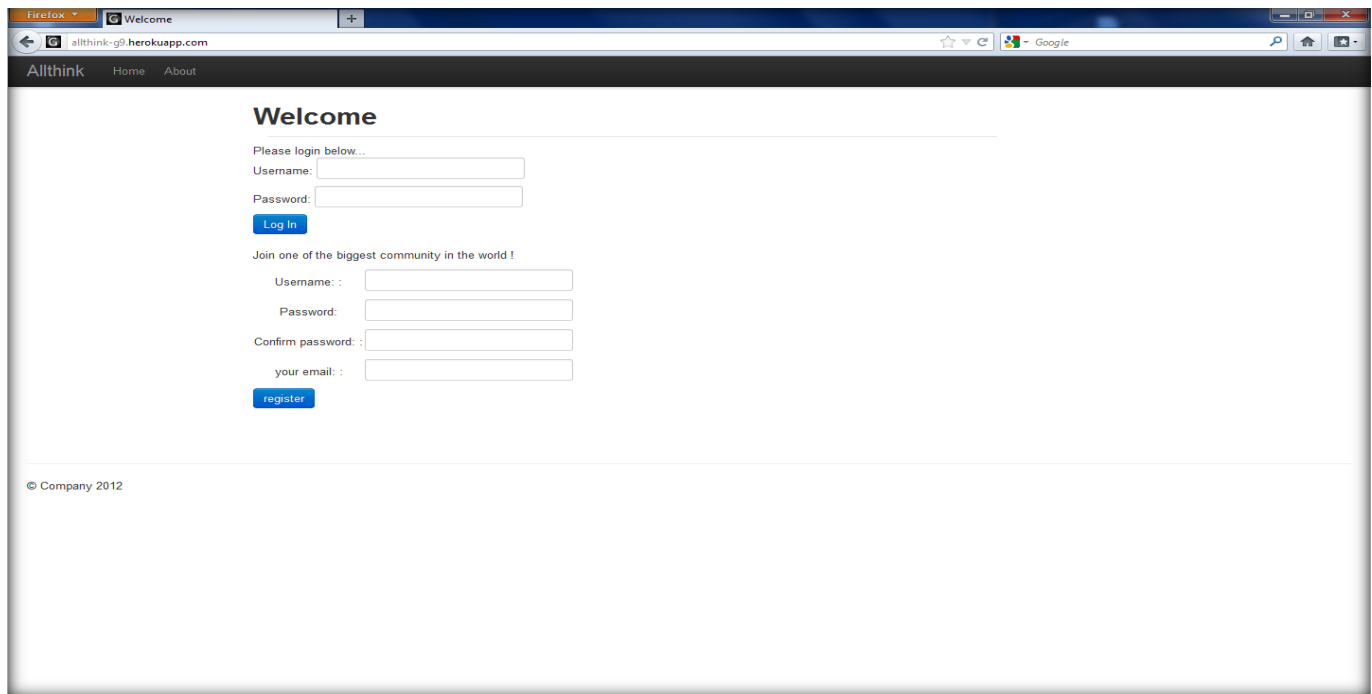
## 3. Application security mechanisms

- Values input into every field are validated before use
- Usernames and passwords are required for access
- Passwords are stored encrypted
- Verification of user email address
- The quality of passwords is checked
- Users must have certificate files on their client machine before they can connect to the server
- Users must have physical security tokens (e.g., hasp, dongle, smartcard, or fingerprint reader)
- Users are given roles that define their permissions

# VII. Design UI

*This UI is easy to use!*

## Login page



The screenshot shows a web browser window with the address bar displaying "allthink-g9.herokuapp.com". The page title is "Welcome". The navigation bar includes "Allthink", "Home", and "About". The main content area features a "Welcome" heading followed by a login section. The login section includes a "Please login below..." prompt, fields for "Username:" and "Password:", and a "Log In" button. Below the login section is a registration section with the text "Join one of the biggest community in the world !", fields for "Username :", "Password:", "Confirm password :", and "your email :", and a "register" button. The footer contains the text "© Company 2012".

Firefox | Welcome | allthink-g9.herokuapp.com | Google

Allthink Home About

### Welcome

Please login below...

Username:

Password:

[Log In](#)

Join one of the biggest community in the world !

Username :

Password:

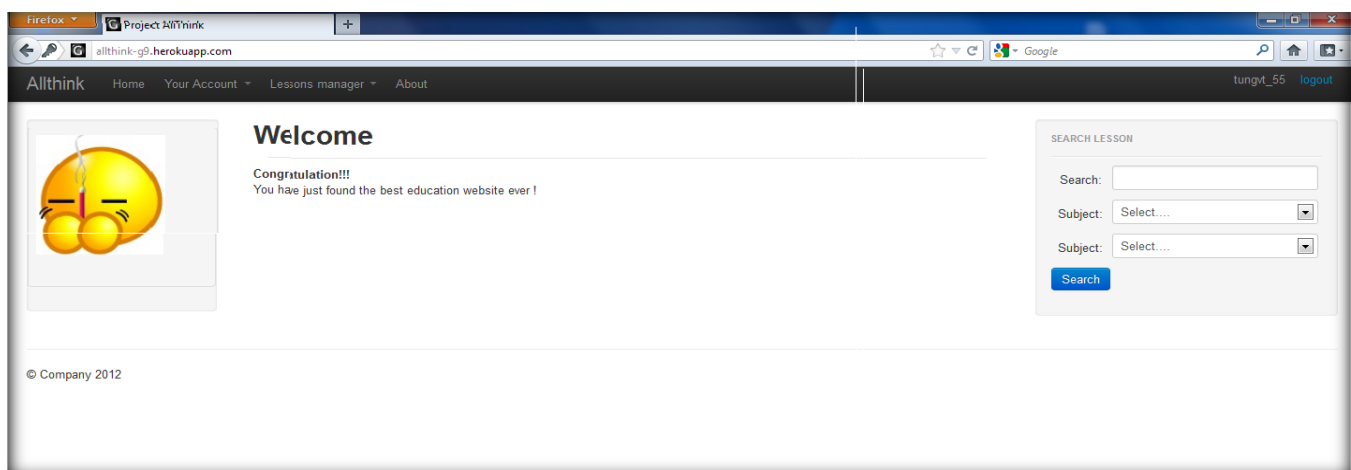
Confirm password :

your email :

[register](#)

© Company 2012

## Main page



The screenshot shows a web browser window with the address bar displaying "allthink-g9.herokuapp.com". The page title is "Project Allthink". The navigation bar includes "Allthink", "Home", "Your Account", "Lessons manager", and "About". The main content area features a "Welcome" heading followed by a congratulatory message: "Congratulation!!! You have just found the best education website ever !". To the left of the message is a cartoon illustration of a yellow character with a lightbulb for a head. To the right is a "SEARCH LESSON" section with a "Search:" input field, two "Subject:" dropdown menus, and a "Search" button. The footer contains the text "© Company 2012".

Firefox | Project Allthink | allthink-g9.herokuapp.com | Google

Allthink Home Your Account Lessons manager About | tungvt\_55 | logout

### Welcome

Congratulation!!!  
You have just found the best education website ever !

SEARCH LESSON

Search:

Subject:

Subject:


[Search](#)

© Company 2012

### Avatar page

Allthink Home Your Account ▾ Lessons manager ▾ About

## Your Profile



Vu Tien Tung 10020423 K55CA



Information:

Vu Tien Tung  
10020423  
K55CA

Avatar:

### Lesson page

## Your Lessons

 computer science	<input type="button" value="✕"/>
 Math	<input type="button" value="✕"/>

[add new lesson](#)

## VIII. User FAQ

### 1. General Information

- PRODUCT NAME: *ALLTHINK*
- Using *ALLTHINK*
- People want to create their own lessons and share them for another people.

## 2. Download and Install

- *How can I obtain ALLTHINK?*
  - ❖ This is a website. You can access its address by any Internet browser: Internet Explorer, Mozilla Firefox, and Google Chrome.
  - ❖ You need to create a new account to use this website.
- *What do I need to use ALLTHINK?*
  - ❖ System requirements are described in the [release notes](#) file.
  - ❖ System requirements are a Intel-compatible PC can run Internet browser whose version is in the time period from 2009 to now and one of the following operating systems: Windows 98/2000/XP, win7, Mac OSX, Linux.
- *How do I upgrade from an older version of ALLTHINK?*
  - ❖ We usually upgrade website once per month. All essential functions will be reserved; the interface and some errors as well as inconvenient functions will be improved. You just need go to website address and check its modification.

## 3. Getting Started

- *How do I register a topic?*
  - ✓ Of course, you need registry first. If you are ready, you can:
  - ❖ Step one: In main page, you can see list of lesson.
  - ❖ Step two: Choose the lesson you want to enroll and clicking on Join button
  - ❖ Step three: If Creator accept, you can enter this class.

## 4. Troubleshooting

- *How do I do when I lost my password?*
  - ✓ You need to contact admin through number 01688580192 or email [tungvt\\_55@vnu.edu.vn](mailto:tungvt_55@vnu.edu.vn) to request to recover your password.

## 5. Other Questions

- *My question is not on this page. How do I find the answer?*
  - First read the [users' guide](#) and other on-line help. Your question may have already been asked and answered, to find it: search the project [mail archives](#) and [issue tracking system](#). If you still don't find it, you can ask the question on the [users' mailing list](#) or the [developers' mailing list](#) or you can [enter an issue](#).

## IX. Implementation Notes

Type of system:	Website Web service
Programming Language(s):	Python HTML, JavaScript, CSS
Data Storage:	SQL database: SQLite
UI Technologies:	Java Swing HTML, CSS, JavaScript
Security Technologies:	Authentication: Local username and password file Authorization: Access control lists

## X. Release notes

### 1. Introduction

- This document contains the release notes for ALLTHINK version 2.1. The following sections describe the release in detail and provide late-breaking or other information that supplements the main documentation.
- This is a developer release for internal evaluation only. Please report any issues via the internal issue tracker.
- This is an early access release for wide evaluation and usage. Your feedback is important to us, please help us make this the best product possible. Keep in mind that we are continuing to work on ALLTHINK and things may change in the future.

### 2. What's New?

- Increase new features such as: registering and changing password, etc.
- We composed streamlined build process
- Roughly doubled unit test coverage has been released
- Many improvements to the product's quality, reliability, ease of use, and performance. See "Recent Changes" below for details.

### 3. Installation and Upgrade Notes

- Installation
  - See the Installation for full details. Please note that in this release, ...
  - **IMPORTANT:** You must completely uninstall any previous "developer release" or "early access" version of this product before installing this release.
  - Manifest
 

This release consists of the following items:

Release notes (this file)

Installation instructions / Quick start guide

User guide

Product source code and build instructions

- Minimum System Requirements

System Processor:	<b>800MHz</b>
System Memory:	256MB
Free Disk Space:	100MB
Operating System:	Windows 2000, Windows XP, Win7, Mac OS X, Linux (kernel 2.4)
Networking:	Internet access
Existing Software:	Apache Tomcat.

- Version Compatibility
  - ❖ You should remove the file using in the previous version before trying to install version 2.1 to guarantee the software run smoothly.

#### 4. Recent Changes

- Fix two login windows to only one.
- Add more features such as: registering and posting new course.
- More compatible and easy-used improvement, showing in the simple interface, etc...

## XI. Test Case

### *1. Login-1: Normal User Login*

Purpose:	<b>Test that users can log in with the proper username and their password.</b>
Prereq:	User is not already logged in. User test user exists, and account is in good standing.
Test Data:	username = {tungvt_55, vutientung010192@gmail.com, empty} password = {123456, 456789, empty}



Steps:	<ol style="list-style-type: none"> <li>1. visit login page</li> <li>2. enter username</li> <li>3. enter password</li> <li>4. click login</li> <li>5. come to main page</li> <li>6. view the information about account and lesson</li> </ol>
Notes and Questions:	<ul style="list-style-type: none"> <li>• Login to main page successfully</li> <li>• This works with both students and advisors</li> </ul>

## ***2. Login-2: Locked-out User Login***

Purpose:	<b>Test that a user who has been locked out by a moderator, cannot log in, They should see a message indicating that they were locked out.</b>
Prereq:	User is not already logged in. User testuser2 exists, and has been locked out
Test Data:	usernameOrEmail = {testuser2, testuser2@nospam.com} password = { valid }
Steps:	<ol style="list-style-type: none"> <li>1. visit LoginPage</li> <li>2. enter usernameOrEmail</li> <li>3. enter password</li> <li>4. click Login</li> <li>5. see LoginPage</li> <li>6. verify warning message is the locked-out message</li> </ol>
Notes and Questions:	<ul style="list-style-type: none"> <li>• Does this work without browser cookies?</li> </ul>

## ***3. Viewing the Lesson***

Purpose:	<b>Test that a user can view the available courses and then can register for courses if they are students or can add new courses if they are advisors.</b>
Prereq:	User already in database
Test Data:	username = { tungvt_55 } VAR = { 123456 }
Steps:	<ol style="list-style-type: none"> <li>1. login to the main page</li> <li>2. click on the manager lesson category in the top</li> <li>3. verify whether seeing the lesson</li> </ol>
Notes and Questions:	<ul style="list-style-type: none"> <li>• NOTE: both kinds of user can view</li> </ul>

## ***4. Accept the registration and add new lesson***

Purpose:	Test that anyone can registry and add new lesson
Prereq:	
Test Data:	VAR = { tungvt_55 } VAR = { 123456 }

Steps:	<ul style="list-style-type: none"> <li>➤ Access to web-site</li> <li>➤ Fill the user and password then click Register</li> <li>➤ Login into main page</li> <li>➤ Click button manager lesson</li> <li>➤ Click add lesson</li> </ul>
Notes and Questions:	<ul style="list-style-type: none"> <li>• Easy to registration and add new lesson</li> </ul>

## ***5. Changing avatar***

Purpose:	Test that user can change avatar easily
Prereq:	Data is available in server
Test Data:	VAR = { tungvt_55 } VAR = { 123456 }
Steps:	<ul style="list-style-type: none"> <li>➤ Access to web-site</li> <li>➤ Fill the user and password then click Login</li> <li>➤ In main page click button Your account</li> <li>➤ Click Brower to get avatar</li> <li>➤ Click Save to finish.</li> </ul>
Notes and Questions:	<ul style="list-style-type: none"> <li>• Easy to change avatar</li> </ul>