UNIVERSITY OF BIRMINGHAM

Software Engineering II (26426)
Group Coursework

Group: AUB
Project: UOB GO!

Group Details

Programme:	BSc		
Members:	1827673	Wing Shan Tam	wst773@student.bham.ac.uk
	1831535	Hiu Yin Tang	hyt735@student.bham.ac.uk
	1855119	Haotian Yan	hxy719@student.bham.ac.uk
	1355330	Inzamam Hussain	ixh330@student.bham.ac.uk

Table of Contents

1. Introduction

1.1. Task

The University of Birmingham will be starting a new campus in Dubai and we have been tasked to provide an innovative interactive application to integrate campuses and allow students, staff and visitor's ways to interact and engage whilst providing a cultural, social and learning experience. Along with integration in the proposed Birmingham-Dubai Digital Wall(s) (2x2 meter screens that will be situated in different focal points).

The proposed application: UOB GO!

A Mobile game to allow users to socialize, learn and compete. The game will feature two maps consisting of both the Birmingham and Dubai campuses respectively. All students and faculty will be registered as players using the UOB database and their UOB ID. Players will be split into two teams respectively, representing their campuses.

The Aim of the game will be for both campuses to compete to see which team can own and hold the most buildings at the end of the academic year. Visitors will be able to play the game for a 24-hour window and complete specific objectives. The digital walls will show campus maps, team scores, building owners, top 5 players and campus news and events.

1.2. Scope

The scope of the system is limited to the design of the Mobile application and its integration to the Birmingham-Dubai Digital Wall.

1.2.1. Mobile

The mobile application is the main platform for players to participate in the game. It will be where players complete tasks such as clear stages within buildings to gain experience and earn rewards, for example: equipment and coins. Stages will allow players to answer questions and if successful they will receive the aforementioned rewards and experience. Players experience will allow them to level up and improve their stats. Players will also be able to equip equipment to improve their stats. Equipment can be obtained as a reward or purchased from the in-game shop using the game coins.

Players can challenge building's owners for ownership. The winner is determined by the player and building owner stats. At the end of the academic year, the team that owns the most buildings will win.

Players will also be able to: chat with other players in game; send, accept and reject friend requests. Once players are friends they will be able to chat one-to-one. We also plan to allow buildings to be explored using external Visual Reality (VR) services that our system will call upon.

1.2.2. Digital Wall

The Birmingham-Dubai Digital Wall will display real time data for the game such as:

1. Campus Maps and Building ownerships
2. Team score Scale bar
3. Top Five players
4. Campus news and events

1.3. Assumptions

1. The system will follow university rules & regulations.
2. The system will use the universities network/database infrastructure.
3. The system will have access to UOB ID database to validate and create players.
4. The system will have access to a player’s timetable.
5. The system will scale team scores based on team size.
6. The system will assign a player to a team based on the player’s information.
7. The system will assign random difficulty to a stage.
8. The system will assure players can only have one account and be on one team.
9. The system will assure visitor accounts will be deleted after 24hour period
10. The system will assure each building in the map belongs to at least one department.
11. The system will assure each building in the map can only have one owner.
12. The system’s default language is English.

2. Requirements

2.1. Functional Requirements

2.1.1. Mobile

FR-M1 Map

FR-M1.1 The system must provide two campus maps (Birmingham and Dubai respectively).

FR-M1.2 The system must allow player to switch between campus maps.

FR-M1.3 The system should display the name of a building owner.

FR-M2 Building

FR-M2.1 The system must provide several stages in each building.

FR-M2.2 The system must allow players to challenge a stage.

FR-M2.3 The system should allow user to explore the building's interior using VR.

FR-M2.4 The system should allow players to view events in a particular building for upcoming 24 hours.

FR-M3 Stage

FR-M3.1 The system must use player’s details to scale difficulty.

FR-M3.2 The system must allow players to receive random rewards after completing a stage.

FR-M3.3 The system should increase rewards if a stage is in a building in the opposing team's campus.

FR-M4 Combat

FR-M4.1 The system must allow players to challenge building owner.

FR-M4.2 The system must allow players to become the building owner if the player wins combat.

FR-M4.3 The system must use players and building owner’s stats to determine winner (new owner).

FR-M5 Player

FR-M5.1 The system must allow players to view player information.

Player information = name, Player ID, avatar, team, department, level, coins, experience, bag, equipped equipment, stats

Stats = (health point (HP), attack (ATK), defence (DEF), equipment (EQP))

FR-M5.2 The system must allow players to change their avatar.

FR-M5.3 The system must allow players to change their equipment.

FR-M5.4 The system must update players stats based on experience.

FR-M6 Login

FR-M6.1 The system must allow players to login/logout the game.

FR-M6.2 The system must allow visitors to login/logout using their 24-hour temporary account.

FR-M7 Chat

FR-M7.1 The system must provide a chat channel for players to chat with other players in game.

FR-M8 Friends

FR-M8.1 The system should allow players to search friends using their Player ID.
FR-M8.2 The system should allow players to send friend request to another player.
FR-M8.3 The system should allow players to accept or reject friend’s request.
FR-M8.4 The system should allow players to block players.
FR-M8.5 The system should allow players to view friends list.
FR-M8.6 The system should provide a one-to-one chat channel for players to chat with friends in their friends list.

FR-M9 Setting

FR-M9.1 The system should allow players to select game language.
FR-M9.2 The system should allow players to set notification time before upcoming events.
FR-M9.3 The system should allow players to activate Global Positioning System (GPS) location.
FR-M9.4 The system should allow players to disable GPS location.

FR-M10 Sign In

FR-M10.1 The system should allow players to sign in to a building during an event in their timetable which is at said building.
FR-M10.2 The system should provide players rewards for signing in.
FR-M10.3 The system should use GPS to obtain player coordinates.

FR-M11 Visitor

FR-M11.1 The system should allow visitor to create a 24-hour temporary account.
FR-M11.2 The system should provide several visitor specific stages for visitors only.
FR-M11.3 The system should allow visitor to complete visitor’s specific missions.

FR-M12 Shop

FR-M12.1 The system should allow players to buy the equipment from the game shop using in game coins.

2.1.2. Birmingham-Dubai Digital Wall

FR-W1 Map

FR-W1.1 The system must allow everyone to view the name of a building owner.

FR-W2 Scale Bar

FR-W2.1 The system must allow everyone to view the score scale bar

FR-W3 Leader Board

FR-W3.1 The system must allow everyone to view the top 5 players in each campus.

FR-W4 Noticeboard

FR-W4.1 The system should provide a Noticeboard to allow everyone to the view campus news and special events on the Noticeboard.

2.2. Non-Functional

2.2.1. Product Requirements

NR-P1 Security

NR-P1.1 The system must restrict access permissions for the system data and may only be changed by the system administrators.

NR-P1.2 The system must not allow players to update their academic information and any such attempt shall be reported to the security administrator.

NR-P1.3 The system must ensure passwords shall never be viewable at point of entry or at any other time.

NR-P1.4 The system must encrypt the login password when sending data to database authentication to guarantee the rights of the players.

NR-P1.5 The system should record each unsuccessful attempt by the player to access an item data on an audit trail.

NR-P2 Availability

NR-P2.1 The system must be in real-time.

NR-P2.2 The system must be available for use 24/7.

NR-P2.3 The system should achieve 99% up time.

NR-P2.4 The system should present players with a notification that the system is unavailable unless the system is non-operational.

NR-P3 Efficiency

NR-P3.1 The system must ensure the restart cycle must execute completely in less than 60 seconds.

NR-P3.2 The system must be able to process a request in 1 second or less.

NR-P3.3 The system must accommodate 70% simultaneous registered players during peak loads.

NR-P3.3 The system must accommodate 90% simultaneous registered players during off peak.

NR-P3.4 The system must ensure any interface between a player and the system shall have a maximum response time of 5 seconds.

NR-P3.5 The system should produce a storage capacity warning when 80% of capacity threshold is exceeded and issues an additional warning after every 5% threshold increments.

NR-P4 Reliability

NR-P4.1 The system must ensure that the account update process should roll back all related updates if any update submissions fails.

NR-P4.2 The system must ensure that the data transmission process confirms that the receiving terminal is in a ready state before transmission begins.

NR-P4.3 The system must check the dialog to ensure the legality of information.

NR-P4.4 The system must have a relatively accurate map for both campuses and its respective buildings

NR-P4.5 The system must disable a player's login for 5 minutes after 5 unsuccessful login attempts.

NR-P5 Usability

NR-P5.1 The system must be able to be used by the general public without training.

2.2.2. Organizational Requirements

NR-O1 Maintainability

NR-O1.1 The system must maintain a service log and check for system service expiration at system start-up.

NR-O1.2 The system should not be closed for maintenance more than once in a 24-hour period.

NR-O2 Flexibility

NR-O2.1 The structure of database should have multi-language support without the need for additional components.

NR-O2.2 The system should ensure every piece of text that players might see must be modifiable without changing source code.

2.2.3. External Requirements

NR-E1 Install Ability

NR-E1.1 The system must be possible to upgrade from any previous version to the main system when a new version of the main system is released.

NR-E1.2 The system must not modify existing configuration values when upgrades are installed.

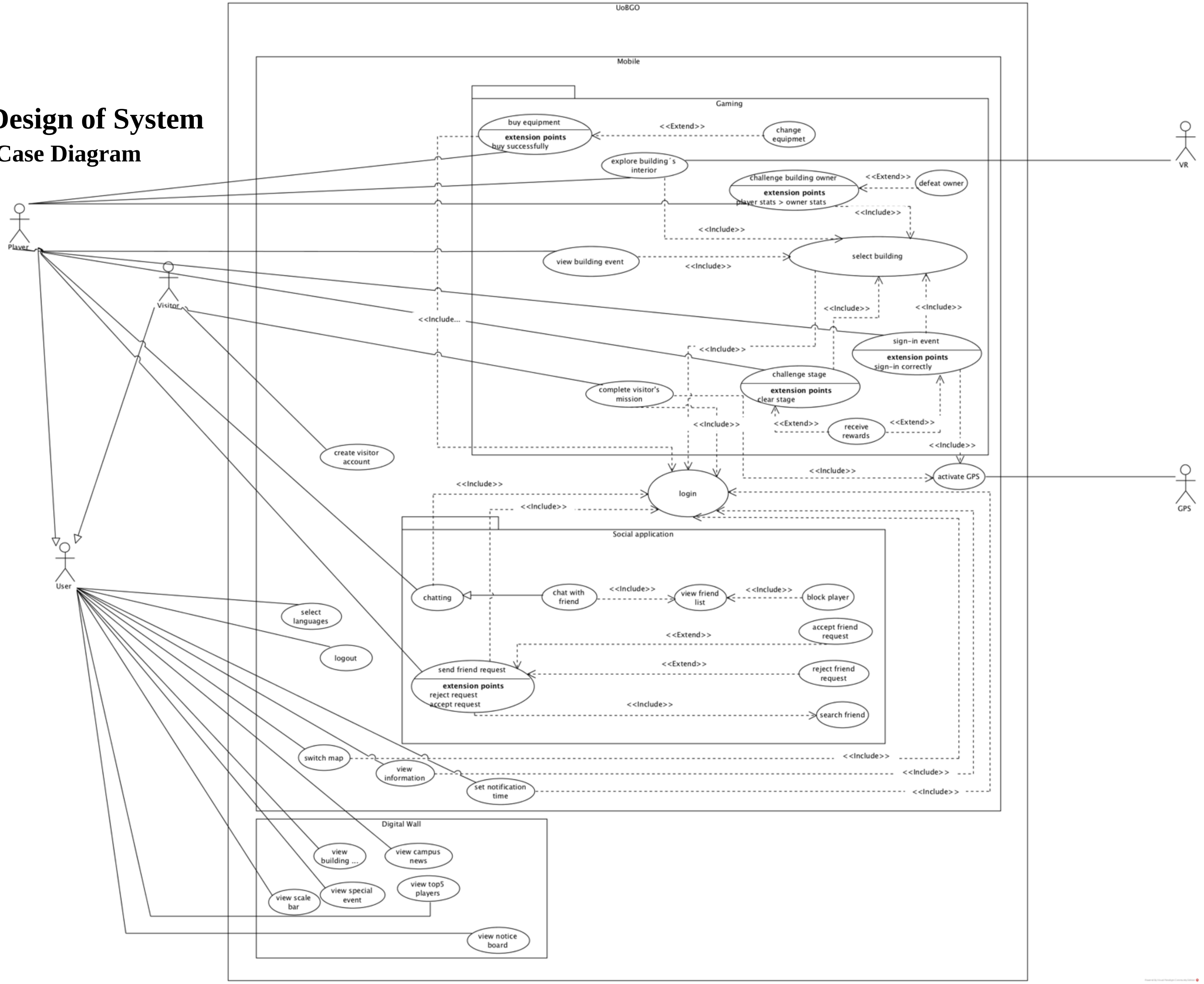
NR-E2 Portability

NR-E2.1 The system should ensure all timestamps recorded should be Coordinated Universal Time (UTC) when permanently stored.

NR-E2.2 The system should ensure the time zone shall be obvious to the user whenever a time element is displayed.

3. UML Design of System

3.1. Use Case Diagram



3.2. Use Cases

Three Use cases have been selected for further documenting.

1. Challenge Building Owner 2. Challenge stage 3. Sign in Event

3.2.1. Challenge Building Owner

Actors	Player
Scenario	The player named James has chosen to challenge the building owner in the game system. He is required to login into the system using his Player ID, which is his UOB ID, and password. After login in successfully, the system displays the Birmingham campus map. He selects the CS building on the map. The system prompts the options list. He chooses the option "Challenge building owner". The system compares the stats of both James and the current building owner. The system determines that James has higher stats than the current building owner. The system shows the combat result and updates the information of building owner.
Pre-condition	1. The player must have an account in the system. 2. The player has selected a particular building. 3. There is already a building owner existing.
Flow of Events	1. Include (Login account). 1.1. The player enters the Player ID and password. The system verifies the login ID and password and display map. 2. Include (Select building) 2.1. The player selects a building on the map. The system displays an options list that available for selecting. 3. The player selects "Challenge building owner". 4. The system compares the stats (including health point (HP), attack (ATK), defence (DEF) and equipment (EQP)) of the player and the building owner. 5. The system determines that he has higher stats than the current building owner. 6. The system shows the combat result on the screen. 7. The system updates the information of building owner.
Alternative flows and exceptions	1. In step 1, if the player has entered wrong login information for five times, his or her account will be locked for 5 minutes. 2. In step 5, if the player stats is equal or less than the building owner stats, then the player loses.
Post-condition	1. The system updates the new building owner.

3.2.2. Challenge Stage

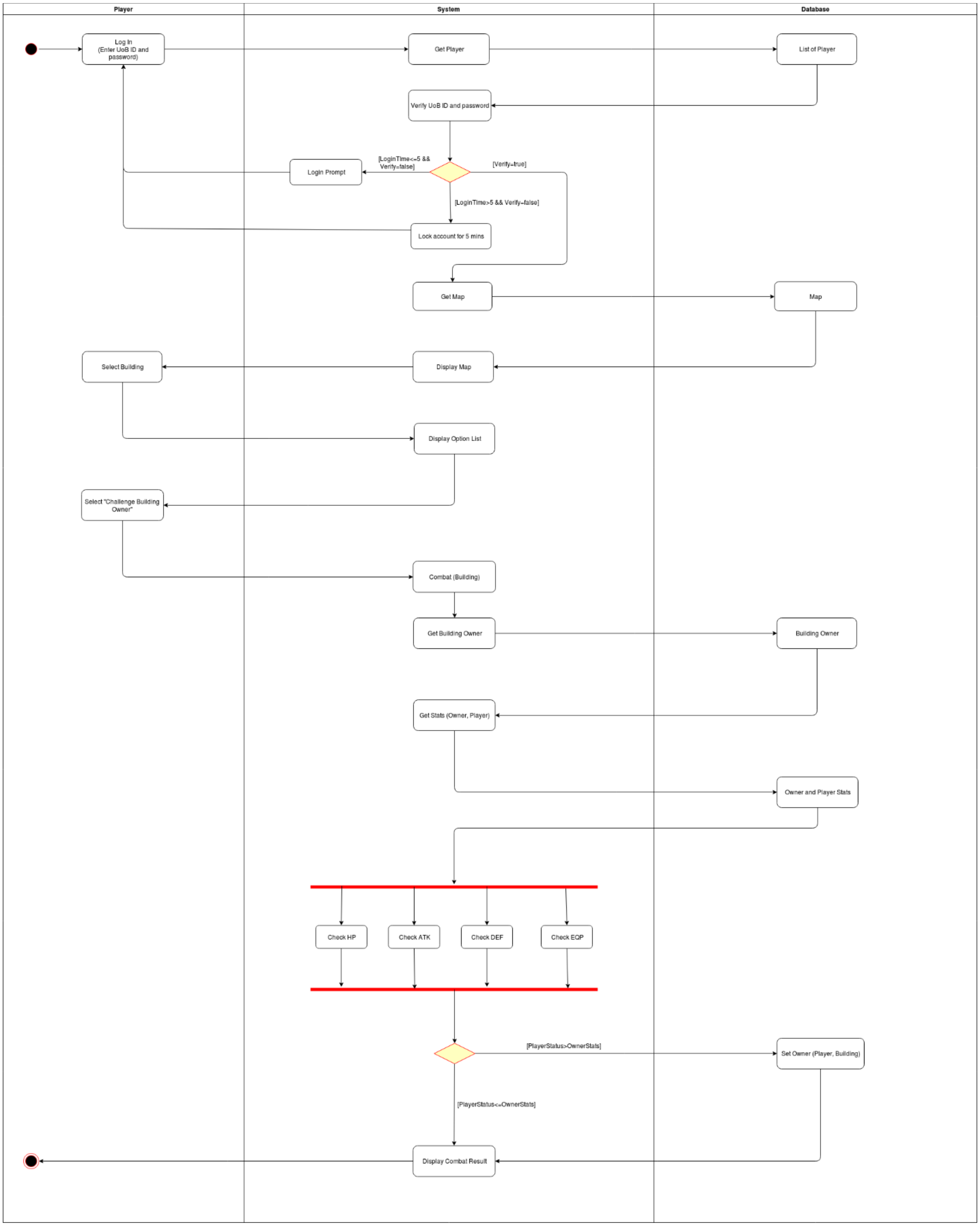
Actors	Player
Scenario	The player named James has chosen to challenge a stage in the game system. He is required to login into the system using his Player ID, which is his UOB ID, and password. After login in successfully, the system displays the Birmingham campus map. The system prompts the options list. He chooses the option "Challenge Stage". The system checks his information and determine the difficulty of the questions. The system picks up the questions based on the difficulty and displays the question. James answer the question and system checks his answer and prompt the next question until James has answered all questions. The system saves the results and provides random rewards.
Pre-condition	1. The player must have an account in the system.
Flow of Events	1. Include (Login account). 1.1. The player enters the UOB ID and password. The system verifies the login ID and password and password and display map. 2. Include (Select building) 2.1. The player selects a building on the map. The system displays a list of options that available for selecting. 3. The player selects "Challenge Stage". The system checks the player information and determine difficulty of the questions. 4. The system picks up random questions based on the difficulty. 5. The system displays a question. 6. The player answers the question. The system checks the answer. 7. The system calculates total scores and saves the results. 8. The system provides random rewards to player.
Alternative flows and exceptions	1. In step 1, if the player has entered wrong login information for five times, his or her account will be locked for 15 minutes. 2. Step 5-6 can be repeated until player answered all questions.
Post-condition	1. The challenge record will be saved in the system and the player will be received the rewards.

3.2.3. Sign in Event

Actors	Player, GPS
Scenario	The player named James has chosen to sign in to an event in the game system. He is required to login into the system using his Player ID, which is his UOB ID, and password. After login in successfully, the system displays the Birmingham campus map. He selects the CS building on the map. The system prompts the options list. He chooses the option "Sign in". When James attends the event in the timetable on time and at the right place, the system provides rewards.
Precondition	1. The player must have an account in the system. 2. The system database contained player's timetable. 3. The player has activated GPS.
Flow of Events	1. Include (Login account). 1.1. The player enters the UOB ID and password. The system verifies the login ID and password and password and display map. 2. Include (Select building) 2.1. The player selects a building on the map. The system displays a list of options that available for selecting. 3. The player selects “Sign In”. The system checks the player location and event in the player’s timetable. 4. The system provides rewards to player.
Alternative flows and exceptions	1. In step 1, if the player has entered wrong login information for five times, his or her account will be locked for 15 minutes. 2. In step 3, if the player does not have event or wrong location, system will inform player and return player to game map.
Post Condition	1. Player has received rewards. 2. Player will no longer be able to sign into this event.

3.3. Activity Diagram

Challenge Building Owner



3.4. Class Analysis

3.4.1. Noun-Verb Analysis

Noun Analysis

Noun/ Noun Phrase	Category	Noun/ Noun Phrase	Category
System	Conceptual	Campus	Attribute
Map	Tangible	Player	Roles played
Location	Attribute	Building Owner	Attribute
Stage	Event	Building	Tangible
Building Interior	Attribute	VR	Other system
Event	Event	Difficulty	Attribute
Reward	Attribute	Team	Attribute
Player Name	Attribute	Player ID	Attribute
Avatar	Attribute	Department	Attribute
Level	Attribute	Coin	Attribute
Experience	Attribute	Equipment	Tangible, Attribute
Equipped equipment	Attribute	Bag	Tangible
Health Point	Attribute	Attack	Attribute
Defence	Attribute	Visitor	Roles played
Chat	Event	Friend	Roles played
Friend List	Conceptual	Language	Attribute
Notification Time	Attribute	GPS	Other system
Sign In	Event	Timetable	Tangible
Mission	Event	Scale Bar	Conceptual
Leader Board	Conceptual	Notice	Event
Module	Conceptual	Stats	Conceptual
Question	Conceptual	Answer	Attribute
Student	Roles played	Staff	Roles played
Department	Tangible		

Candidate Classes

Map	Player
Stage	Building
VR	Event
Equipment	Friend
GPS	Timetable
Notice	Module
Bag	Question
Student	Staff
Department	

Verb Analysis

Verb/ Verb Phrase	**Class**	**Verb/ Verb Phrase**	**Category**
Switch Map	Map	Display player's character	Outside scope
Display the name of a building owner	Building	Challenge a stage	Stage
Explore the building's interior	Building	View events	Event
Receive random rewards	Stage	Scale difficulty	Stage
Challenge building owner	Building	Change avatar	Player
Change equipment	Player	Update player stats	Player
Login	Outside scope	Logout	Outside scope
Chat	Outside scope	Search friends	Player
Send friend request	Friend	Accept or reject friend’s request	Friend
Block players	Friend	Select game language	Player
Set notification time	Player	Activate GPS location	Player
View campus news and special events	Notice	View the scale bar	Building
Sign in to a building	Building	Buy the equipment	Equipment
Obtain player coordinates	Outside scope	create a 24-hour temporary account	Visitor
View the top 5 players	Player		

3.4.2. CRC Cards

4. Player	
Responsibilities	**Collaborators**
Maintain data about particular player information, including operations related to player information.	Map Building Stage Notice Equipment Timetable Bag Equipment Friend Department

Student	
Responsibilities	**Collaborators**
Maintain data about particular student information.	Player

Staff	
Responsibilities	**Collaborators**
Maintain data about particular staff information.	Player

Department	
Responsibilities	**Collaborators**
Maintain data about a department information.	Player Building

Map	
Responsibilities	**Collaborators**
Maintain data about campus map.	Building Player Visitor

Building	
Responsibilities	**Collaborators**
Maintain data about a particular building, including operations related to player option.	Map Player Department Visitor Stage Event

Stage	
Responsibilities	**Collaborators**
Maintain data about a particular stage including challenge and provide rewards.	Building Player Visitor Question

Visitor	
Responsibilities	**Collaborators**
Maintain data about a visitor.	Map Building Stage

Question	
Responsibilities	**Collaborators**
Maintain data about a question for challenging stage.	Stage

Event	
Responsibilities	**Collaborators**
Maintain data about an event in building.	Building

Bag	
Responsibilities	**Collaborators**
Maintain data about a particular player owned equipment.	Player

Equipment	
Responsibilities	**Collaborators**
Maintain data about a particular equipment details which for buying.	Player

Friend	
Responsibilities	**Collaborators**
Maintain data about a particular player’s friend list.	Player

Timetable	
Responsibilities	**Collaborators**
Maintain data about a particular player's timetable.	Player Module

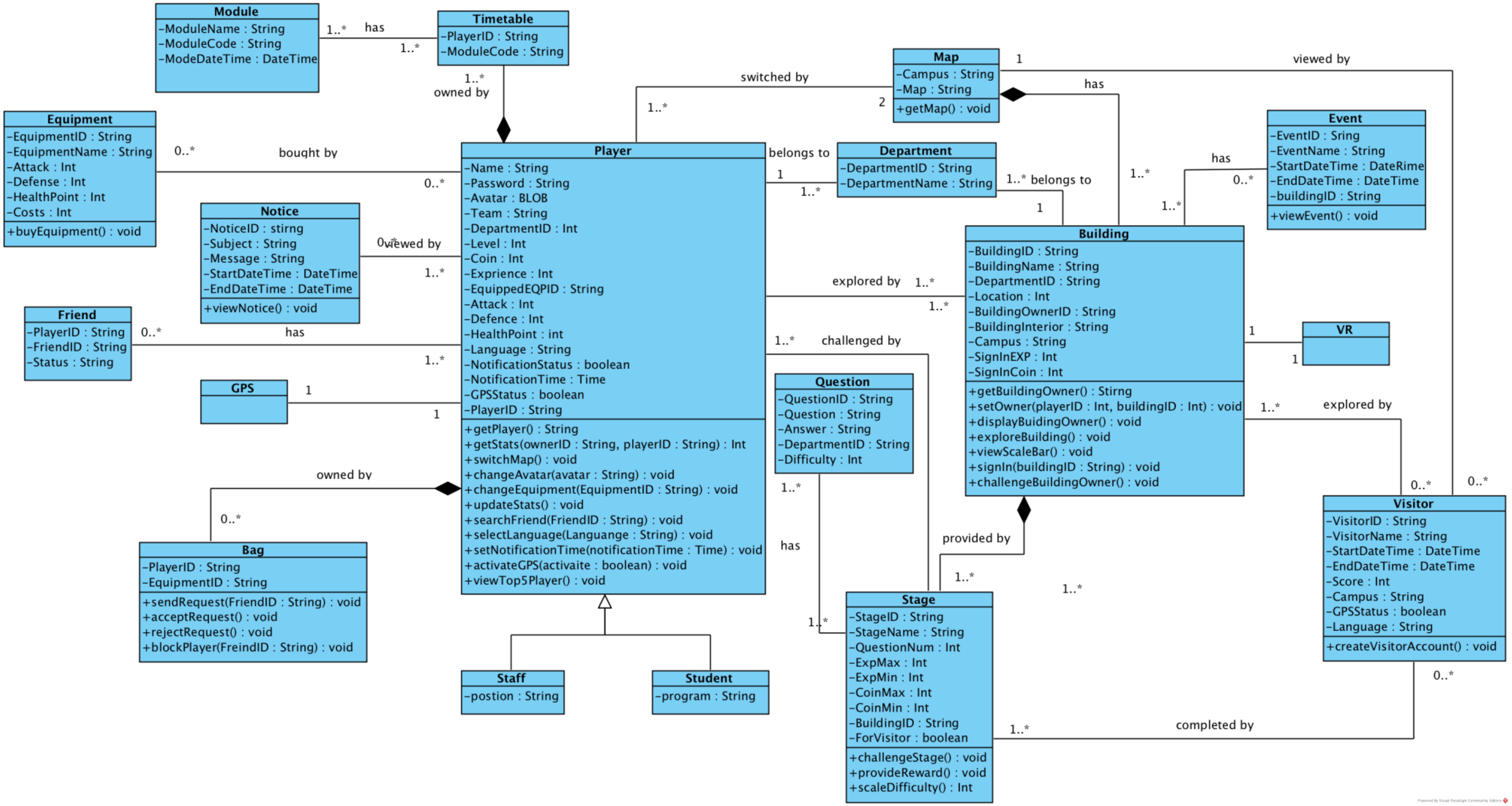
Module	
Responsibilities	**Collaborators**
Maintain data about module details.	Timetable

GPS	
Responsibilities	**Collaborators**
Coordinates the functionality of the display the character on the map.	Player

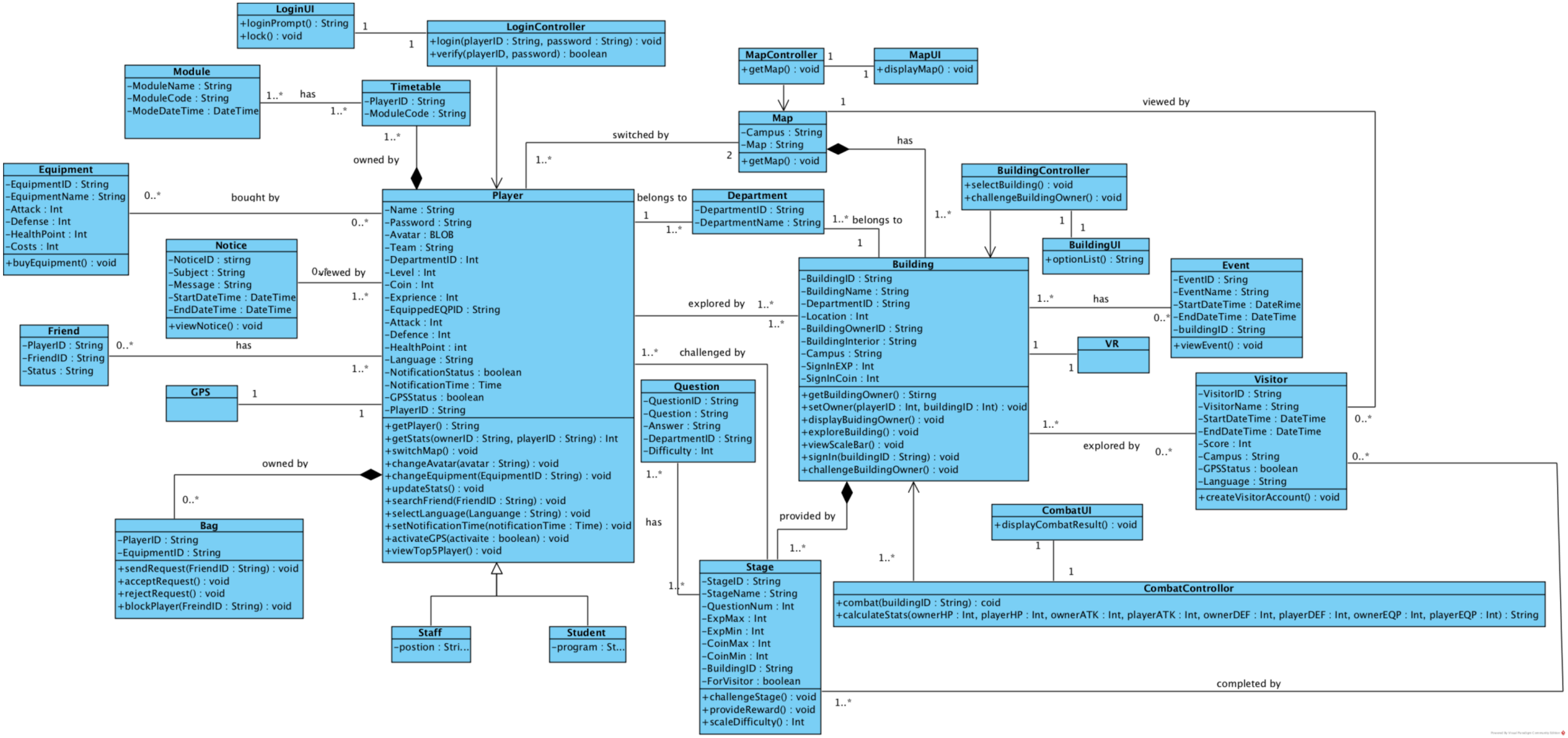
VR	
Responsibilities	**Collaborators**
Coordinates the functionality of the exploring building interior.	Building

3.4.3. Class Diagram

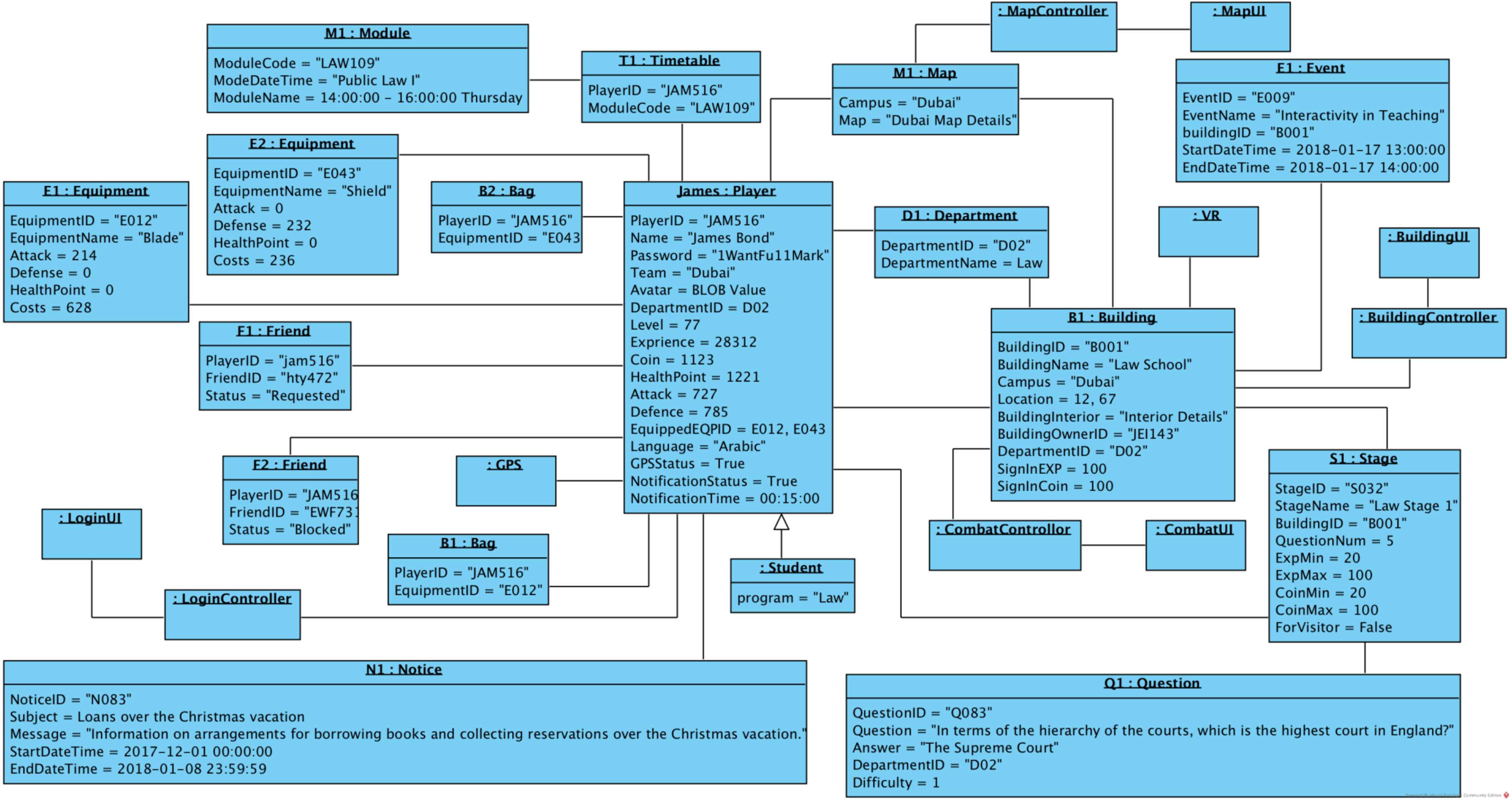
First Cut



Refined

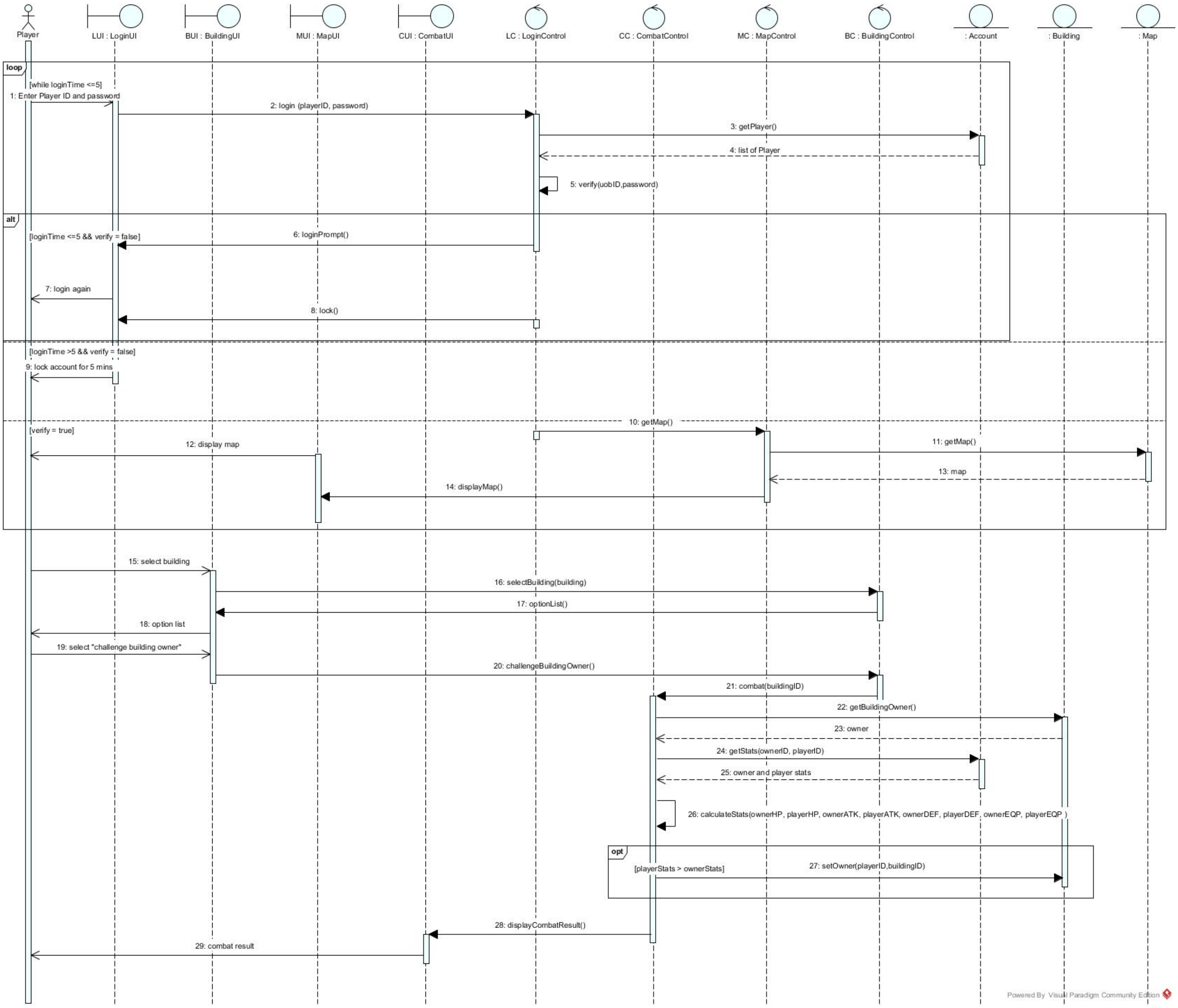

LoginUI
+loginPrompt() : String
+lock() : void
1
1
LoginController
+login(playerID : String, password : String) : void
+verify(playerID, password) : boolean
Module
-ModuleName : String
-ModuleCode : String
-ModeDateTime : DateTime
1..*
has
1..*
Timetable
-PlayerID : String
-ModuleCode : String
1..*
owned by
Equipment
-EquipmentID : String
-EquipmentName : String
-Attack : Int
-Defense : Int
-HealthPoint : Int
-Costs : Int
+buyEquipment() : void
0..*
bought by
0..*
Notice
-NoticeID : stirng
-Subject : String
-Message : String
-StartDateTime : DateTime
-EndDateTime : DateTime
+viewNotice() : void
0..*
viewed by
1..*
Friend
-PlayerID : String
-FriendID : String
-Status : String
0..*
has
1..*
GPS
1
1
owned by
0..*
Bag
-PlayerID : String
-EquipmentID : String
+sendRequest(FriendID : String) : void
+acceptRequest() : void
+rejectRequest() : void
+blockPlayer(FreindID : String) : void
Player
-Name : String
-Password : String
-Avatar : BLOB
-Team : String
-DepartmentID : Int
-Level : Int
-Coin : Int
-Exprience : Int
-EquippedEQPID : String
-Attack : Int
-Defence : Int
-HealthPoint : int
-Language : String
-NotificationStatus : boolean
-NotificationTime : Time
-GPSStatus : boolean
-PlayerID : String
+getPlayer() : String
+getStats(ownerID : String, playerID : String) : Int
+switchMap() : void
+changeAvatar(avatar : String) : void
+changeEquipment(EquipmentID : String) : void
+updateStats() : void
+searchFriend(FriendID : String) : void
+selectLanguage(Languange : String) : void
+setNotificationTime(notificationTime : Time) : void
+activateGPS(activaite : boolean) : void
+viewTop5Player() : void
Staff
-postion : Stri...
Student
-program : St...
switched by
1..*
2
MapController
+getMap() : void
1
1
MapUI
+displayMap() : void
Map
-Campus : String
-Map : String
+getMap() : void
1
viewed by
has
belongs to
1
1..*
Department
-DepartmentID : String
-DepartmentName : String
1..* belongs to
1
1..*
BuildingController
+selectBuilding() : void
+challengeBuildingOwner() : void
1
1
BuildingUI
+optionList() : String
explored by
1..*
1..*
1..*
challenged by
Question
-QuestionID : String
-Question : String
-Answer : String
-DepartmentID : String
-Difficulty : Int
1..*
has
1..*
Building
-BuildingID : String
-BuildingName : String
-DepartmentID : String
-Location : Int
-BuildingOwnerID : String
-BuildingInterior : String
-Campus : String
-SignInEXP : Int
-SignInCoin : Int
+getBuildingOwner() : Stirng
+setOwner(playerID : Int, buildingID : Int) : void
+displayBuidingOwner() : void
+exploreBuilding() : void
+viewScaleBar() : void
+signIn(buildingID : String) : void
+challengeBuildingOwner() : void
1..*
has
0..*
Event
-EventID : Sring
-EventName : String
-StartDateTime : DateRime
-EndDateTime : DateTime
-buildingID : String
+viewEvent() : void
1
VR
1
1..*
explored by
0..*
Visitor
-VisitorID : String
-VisitorName : String
-StartDateTime : DateTime
-EndDateTime : DateTime
-Score : Int
-Campus : String
-GPSStatus : boolean
-Language : String
+createVisitorAccount() : void
0..*
0..*
provided by
1..*
1..*
CombatUI
+displayCombatResult() : void
1
1
Stage
-StageID : String
-StageName : String
-QuestionNum : Int
-ExpMax : Int
-ExpMin : Int
-CoinMax : Int
-CoinMin : Int
-BuildingID : String
-ForVisitor : boolean
+challengeStage() : void
+provideReward() : void
+scaleDifficulty() : Int
CombatControllor
+combat(buildingID : String) : coid
+calculateStats(ownerHP : Int, playerHP : Int, ownerATK : Int, playerATK : Int, ownerDEF : Int, playerDEF : Int, ownerEQP : Int, playerEQP : Int) : String
completed by
1..*

3.5. Object Diagram



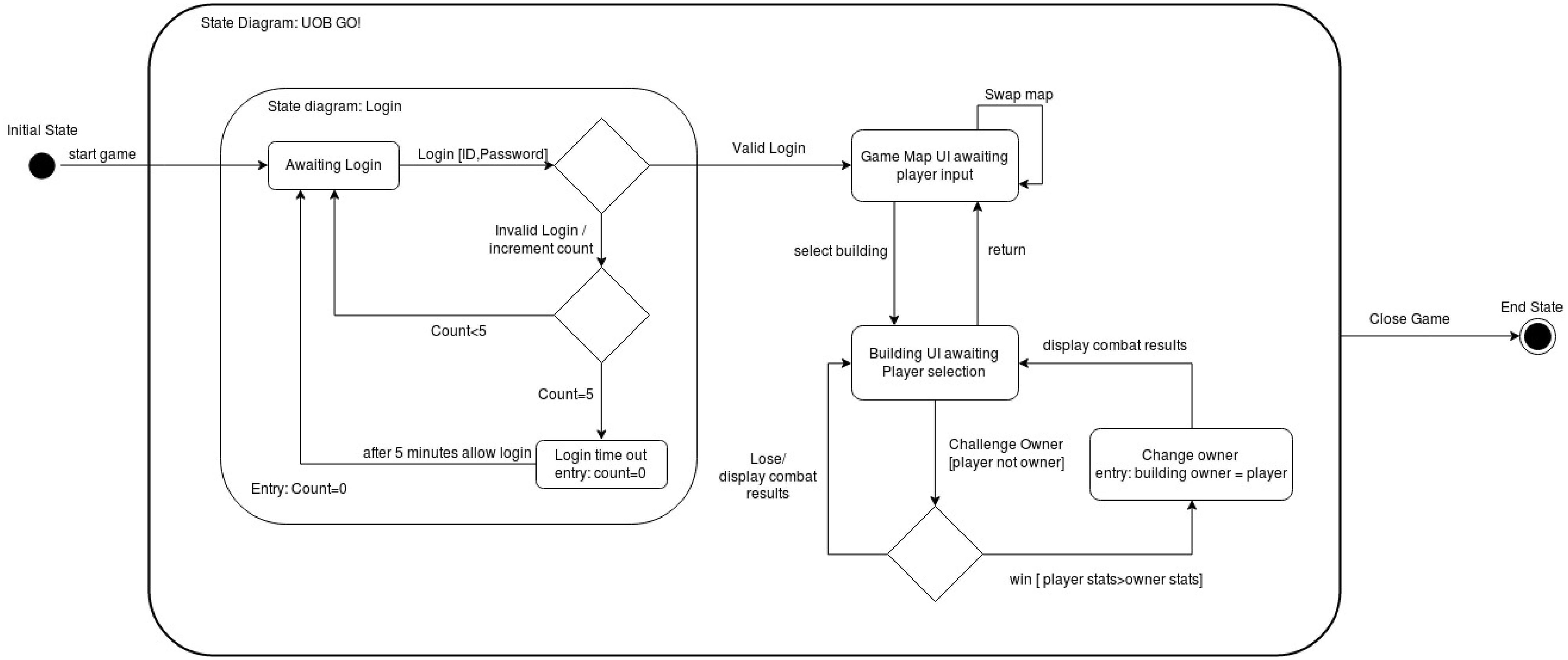
3.6. Sequence Diagram

Challenge Building Owner


Player
LUI : LoginUI
BUI : BuildingUI
MUI : MapUI
CUI : CombatUI
LC : LoginControl
CC : CombatControl
MC : MapControl
BC : BuildingControl
: Account
: Building
: Map
loop
[while loginTime <=5]
1: Enter Player ID and password
2: login (playerID, password)
3: getPlayer()
4: list of Player
5: verify(uobID,password)
alt
[loginTime <=5 && verify = false]
6: loginPrompt()
7: login again
8: lock()
[loginTime >5 && verify = false]
9: lock account for 5 mins
[verify = true]
10: getMap()
11: getMap()
12: display map
13: map
14: displayMap()
15: select building
16: selectBuilding(building)
17: optionList()
18: option list
19: select "challenge building owner"
20: challengeBuildingOwner()
21: combat(buildingID)
22: getBuildingOwner()
23: owner
24: getStats(ownerID, playerID)
25: owner and player stats
26: calculateStats(ownerHP, playerHP, ownerATK, playerATK, ownerDEF, playerDEF, ownerEQP, playerEQP)
opt
[playerStats > ownerStats]
27: setOwner(playerID,buildingID)
28: displayCombatResult()
29: combat result
Powered By Visual Paradigm Community Edition

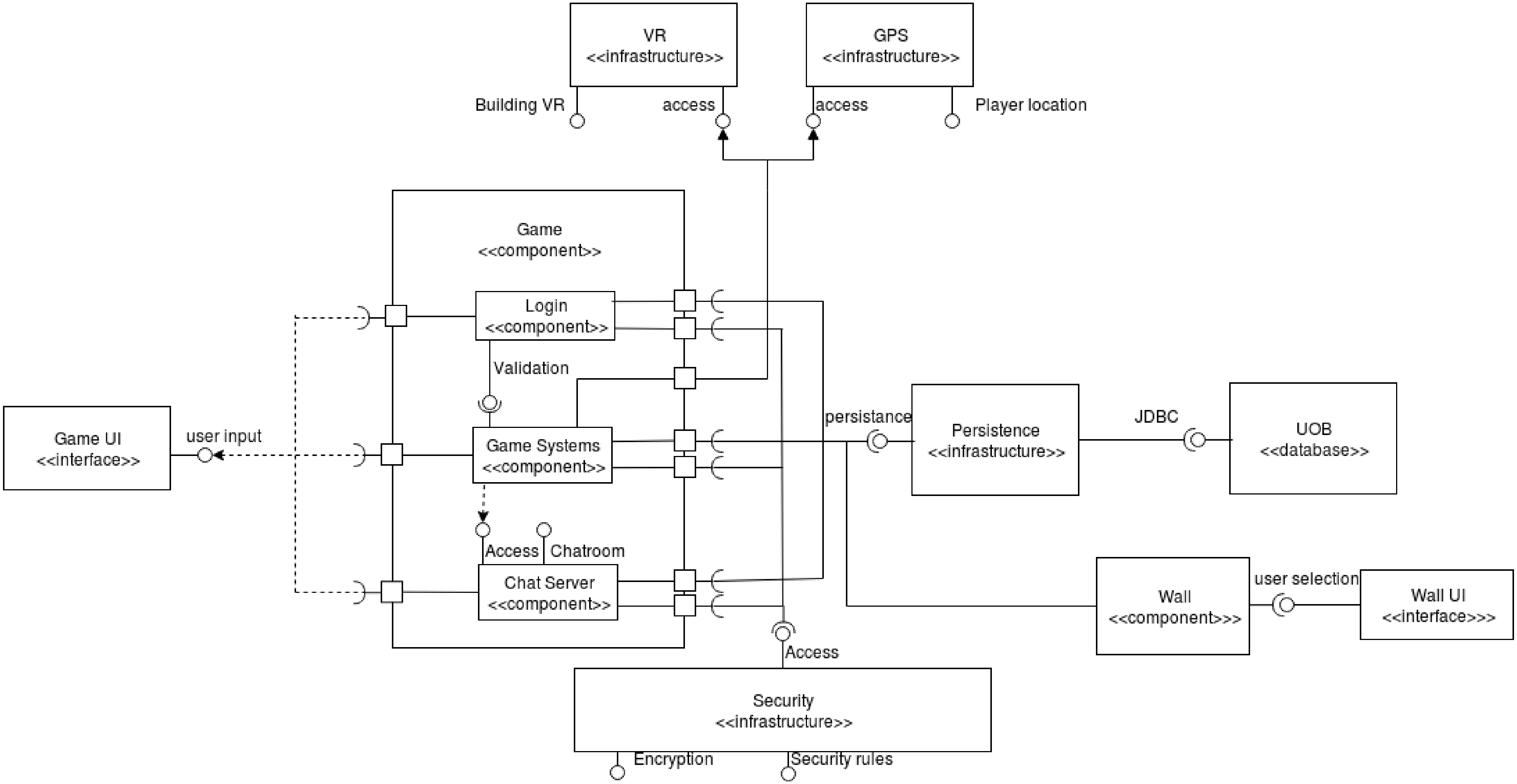
3.7. State Diagram

Challenge Building Owner

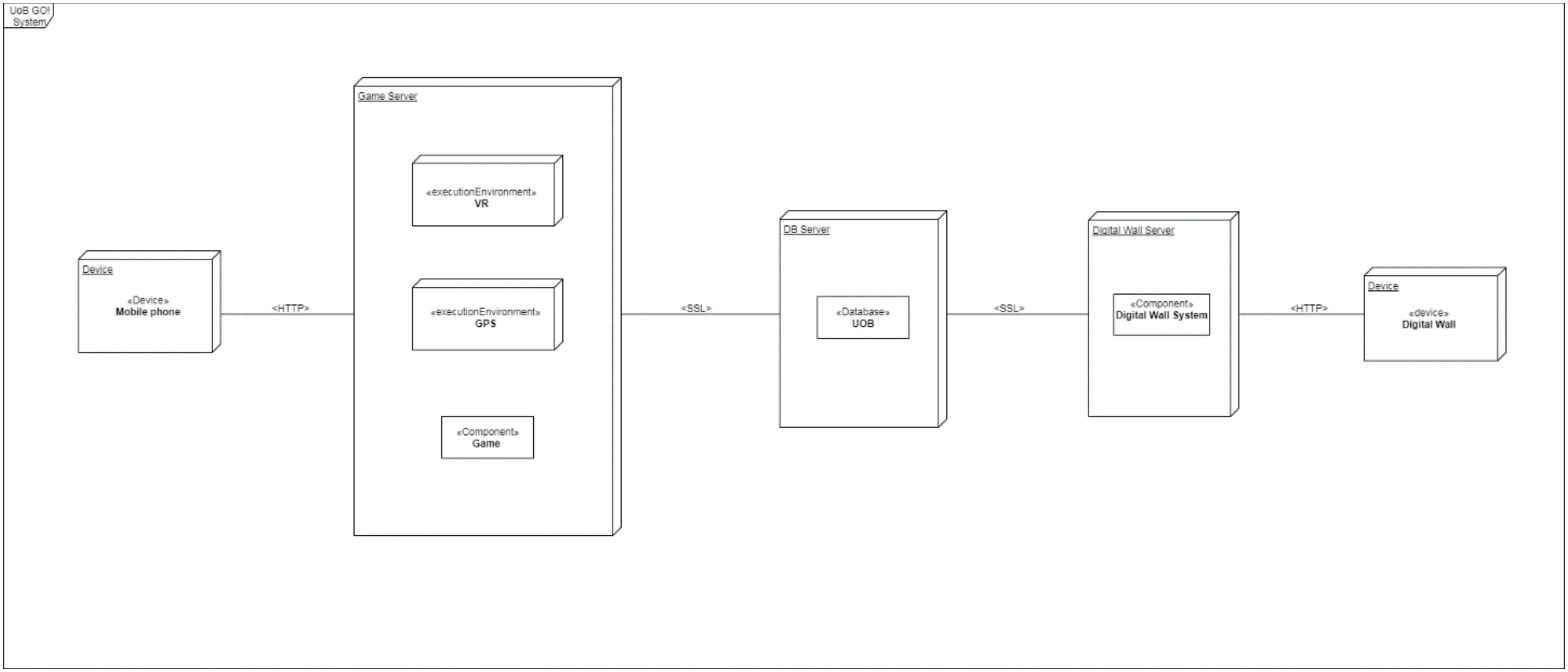

State Diagram: UOB GO!
Initial State
start game
State diagram: Login
Awaiting Login
Login [ID,Password]
Invalid Login / increment count
Count<5
Count=5
after 5 minutes allow login
Login time out
entry: count=0
Entry: Count=0
Valid Login
Swap map
Game Map UI awaiting player input
select building
return
Building UI awaiting Player selection
display combat results
Challenge Owner [player not owner]
Change owner
entry: building owner = player
Lose/ display combat results
win [player stats>owner stats]
Close Game
End State

4. Architecture Designs

4.1. Component Diagram



4.2. Deployment Diagram



4.3. Trade of Analysis

4.3.1 Component diagram

When it came to deciding upon a system architecture and designing the subsequent component diagram we decided to go for a plug and play approach for our components, interfaces, infrastructures and database respectively. We believe this approach allows the system to be easily scalable, maintainable and securable, with potential performance benefits.

All components will be limited to their respective tasks and can easily be replaced when needed with either inhouse developed components our third-party components purchased and implemented/integrated within the existing architecture.

Therefore, the key is ensuring all the communication between components, interfaces, infrastructures and database is not compromised and the relevant information is provided and obtained. This could be a hurdle in the future if the architecture begins to use different types of Languages/protocols we may have be implement wrappers and API's for translation which could affect performance and have potentially security implications.

On the other hand, having a plug and play approach for system architecture and components has allowed us to ensure each component is secured accordingly, depending on its purpose and usage. For example: the login component will be the most secure component within the system as our non-functional requirements indicate the password must never be visible and be encrypted during all communications. Likewise, the chat server infrastructure must meet its own security rules, these security measures/services will be provided by the security infrastructure which will contain all the relevant security functions that other components and infrastructure will be able to access. In our design only login, game systems and chat server have access to this infrastructure.

We have used 4 infrastructures within our design respectively, each of which can be replaced when needed. 2 provide key services, (security and chat server) whilst the other 2 (VR and GPS) provide in game functionality which is not key to the system. Therefore, we can configure the system to prioritise communication with these services accordingly and amend the amount of security applied/applicable for communications to and from said services.

As much as we have embraced the plug and play ideology we have kept all the game systems in one component as we decided it would be very complex to distinguish them all and as they are all are dependent on each other it would be best to package them together within one component labelled game systems. A trade-off for this decision could be that we would not be able to scale, access any one sub system on its own but we are happy with this compromise as having all the systems be in one package allows for less traffic overhead as there are less components communicating and we can secure all the game systems within the one component.

A key principle we required in our architecture was persistency of our system and its communication with the database. The plug and play ideology benefited us here also as having all game sub-systems in one larger game systems component as previously mentioned reduced the number of components that had to communicate with said database.

As mentioned and indicated by our component diagram all communication to the database will be persistent both for the game and the digital wall. Our non-functional requirements have many restraints regarding access, upkeep and communication speeds with the database. Therefore, having a persistent connection to the database allows us to meet all of these requirements head on as being always connected allows the game to be real-time and as far as uptime/maintenance is concerned we are relying on the database (university database) to have similar if not stricter requirements therefore allowing us to meet all these non-functional requirements.

A trade-off for this persistence connection could be reduced performance if the database is not able to handle all the connections but as mentioned previously we are relying on the database to have been designed/able to handle this. If not a benefit of the plug and play architecture is we can always change the database with one more adequate/suited to our needs if need be.

So to conclude we think our architecture design has many benefits with some potential trade-offs but if said trade-offs are severe enough we can always adapt the architecture and replace components where/when necessary to ensure we meet all our requirements.

4.3.2 Deployment diagram

Two devices and two servers are included in this design to ensure the respective realization of mobile system and digital wall system. To reduce the cost of applying two devices and servers, only one database is included in this architecture, which will reduce the cost of maintenance at the same time. However, this set-up would generate a bottleneck within the data processing in the database server because of all the data being processed by one server. In addition, if any problems appeared in the database server, for example some hackers attacked the database server, the whole system would be affected negatively. Thus, there has been a trade off in ease of maintenance and security/accessibility. What’s more, to enhance the security of connection between server and database, SSL (Secure Sockets Layer) is used to ensure the secure data transfer and logins, which would also increase the opportunity for interception of sensitive information in the game system.

Therefore we acknowledge and understand the tradeoffs for our choice of deployment but are confident it is the best approach for deploying our system at launch. In the future based on requirements and needs we could possible add more database servers and/or modify wherever applicable.