Secure-Software Development: International Space Station Project
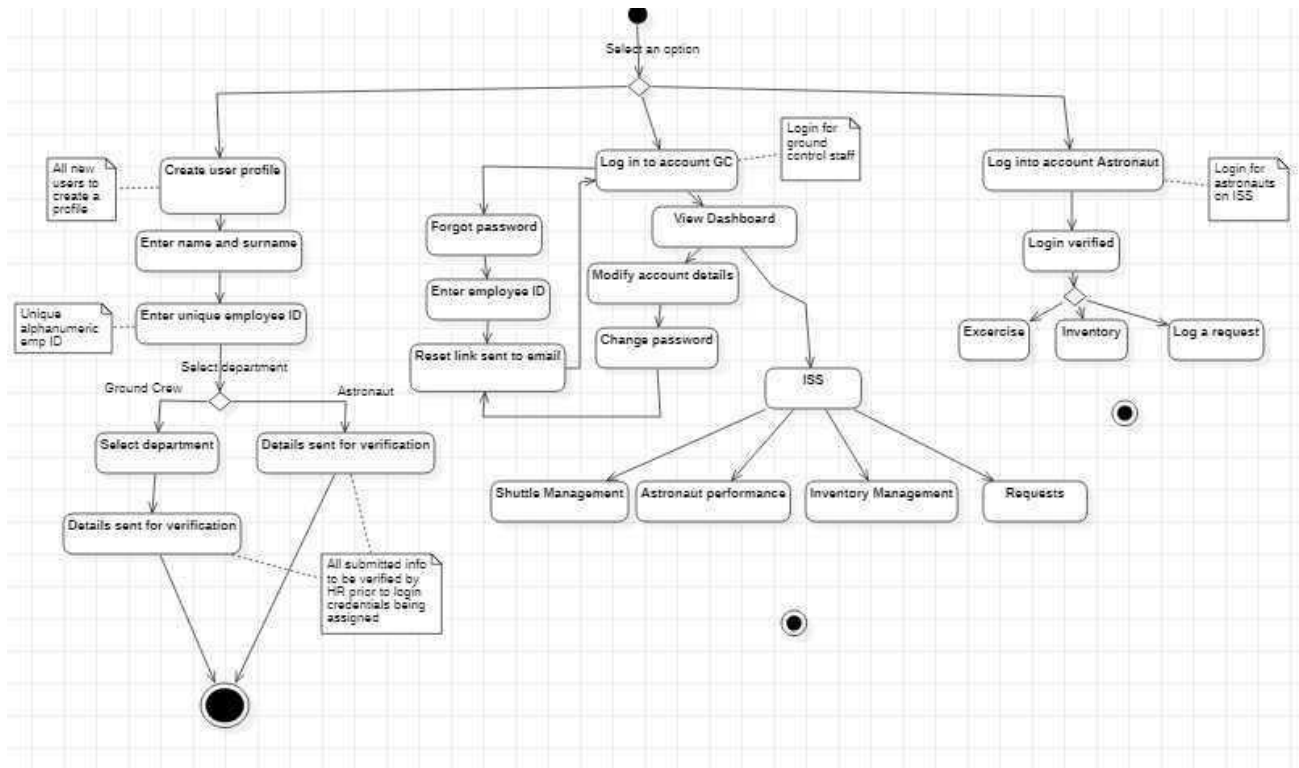
System Requirements

- Our system features two locations, earth and the ISS.
- Our system has two types of users: ground control and astronauts.
- Both users will have a separate interface to access the features specific to them.
- Ground control can create astronaut accounts. Ground control oversees shuttle flights.
- Ground control can add astronauts to space ships and send them to and from the ISS. Astronauts can only access their interface when they are on the ISS.
- The astronaut's interface allows them to access the exercise regimes which monitors their performance.
- The astronauts can see their own performance metrics and the ground control users can oversee all the performance metrics of the astronauts.
- The ISS has an inventory of resources that are required to keep the space station running. The ISS automatically orders resources that are running out. When a resource has been ordered, it is added to the next flight to the space station.
- There are two types of flights, manned and unmanned. Ground control initiates the manned flights, the unmanned flights are scheduled periodically to transfer any pending orders of resources.
- There are a fixed number of docks on the ISS and the flight/docking system ensures that flights are not organized to dock at the same dock.
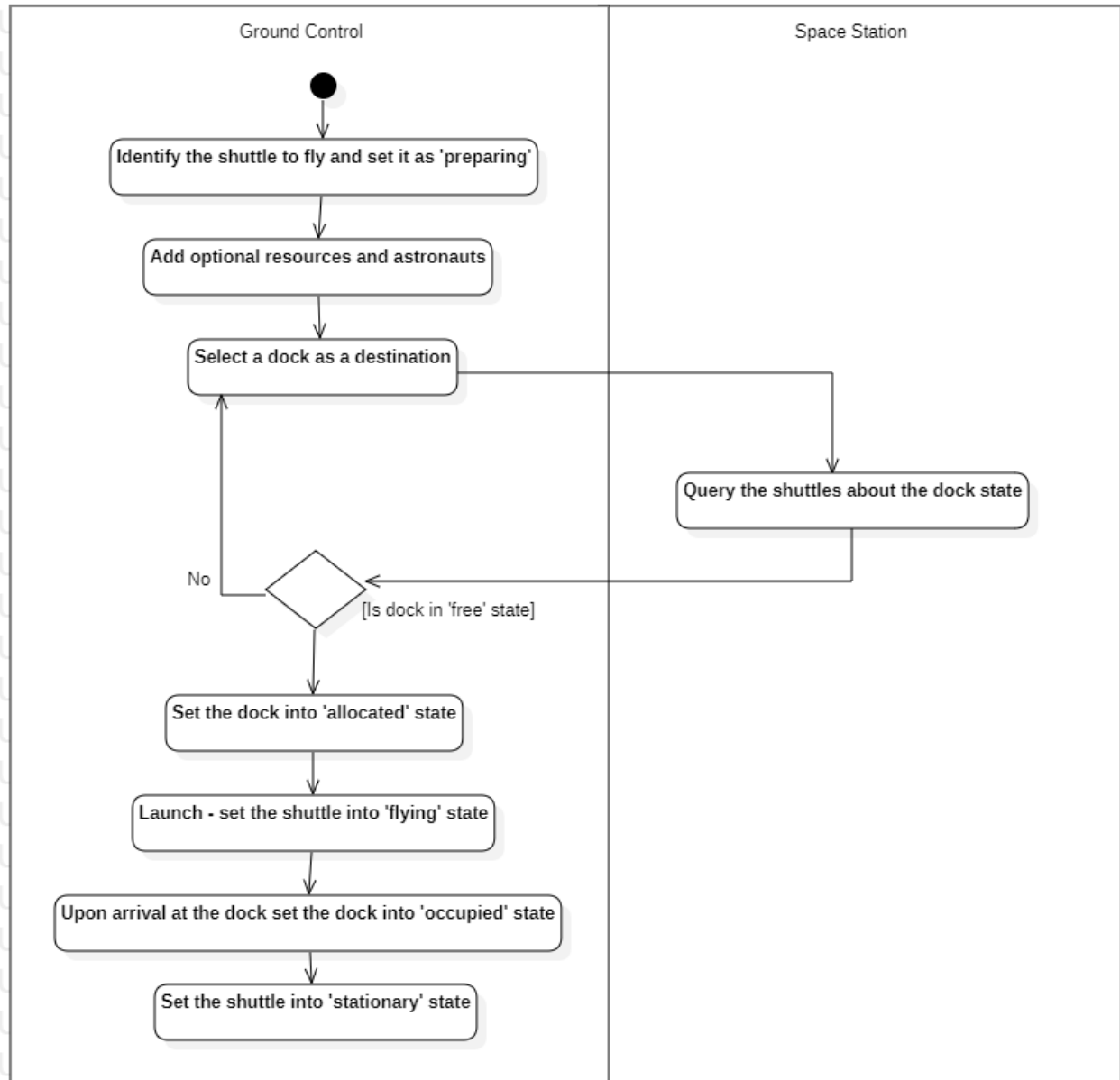
High level design UML diagrams
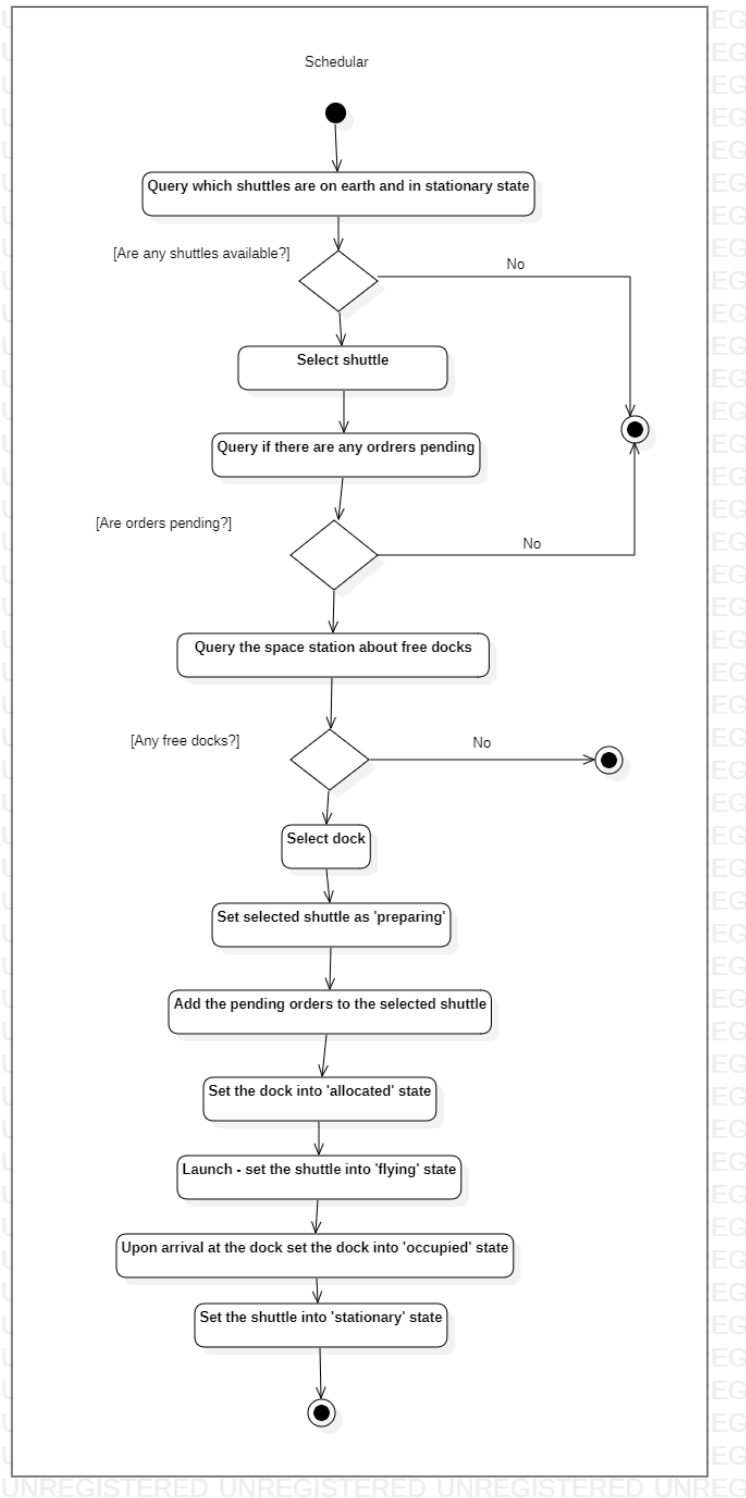
User accounts design:

User accounts activity diagram:

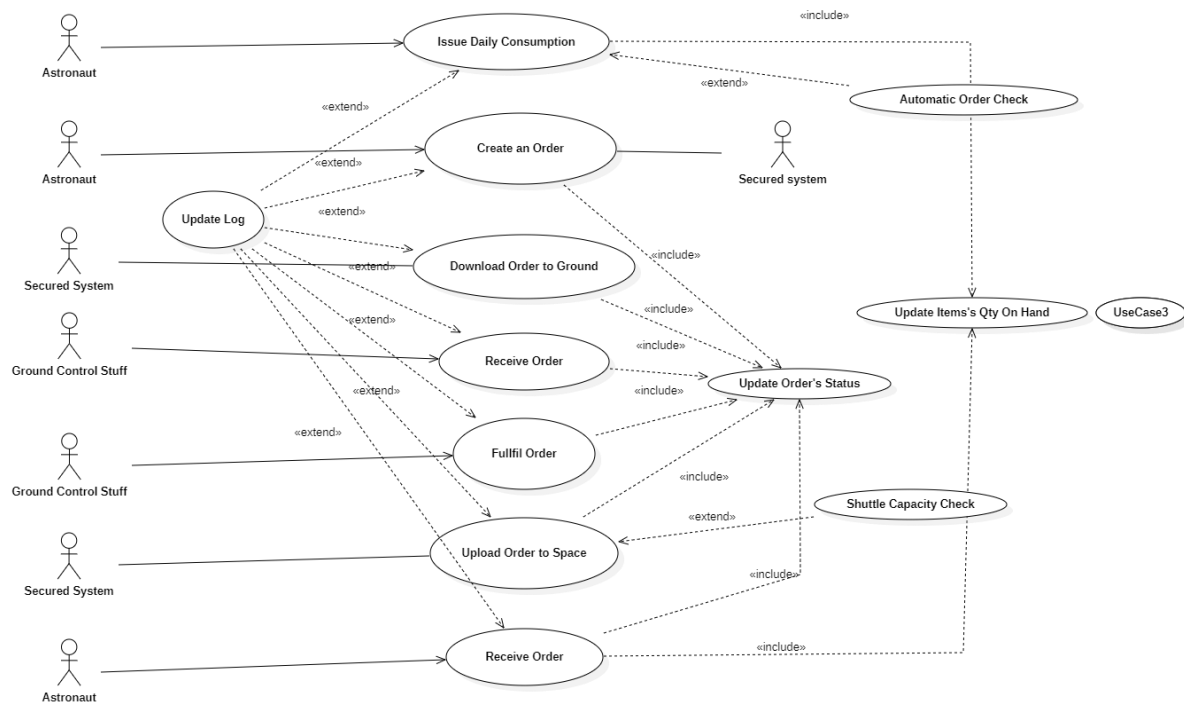## Flight/Docking Design

Maned flight and docking activity diagram:

Un-manned flight and docking diagram:

Schedular

Query which shuttles are on earth and in stationary state

[Are any shuttles available?]   No

Select shuttle

Query if there are any ordrers pending

[Are orders pending?]   No

Query the space station about free docks

[Any free docks?]   No

Select dock

Set selected shuttle as 'preparing'

Add the pending orders to the selected shuttle

Set the dock into 'allocated' state

Launch - set the shuttle into 'flying' state

Upon arrival at the dock set the dock into 'occupied' state

Set the shuttle into 'stationary' state

Resource Management

Use case diagram:

Logistics - Use Case Diagram

Activity Diagram – ISS:

Activity Diagram - Logistics -Space

Login

Check Authorization — No Authorization

{No Robot Check}

{User , Password}

Authorization

{List of Roles & Authorization}

Assign Role

Check Orders to be approved

{A list of orders awaits confirmation}

According to a list of daily consumption

Issue Items

{Min / Max - Qty}

Review Items

Check if Order Arrived

Verify Orders to receive

{List of criteria for approval}

Approve Order

{A list of orders awaits confirmation}

Order Arrived

{Min / Max , Importance}

Evaluate Items On stock

Order Updated Items

Receive Order

Do we need an Order ?

Yes

{List of items}

Create Order

Approve Order

Complete Order

Send to Approval

Approved

Transmit Order To Ground

Logout

Activity Diagram – Ground:

Activity Diagram - Logistics - Ground
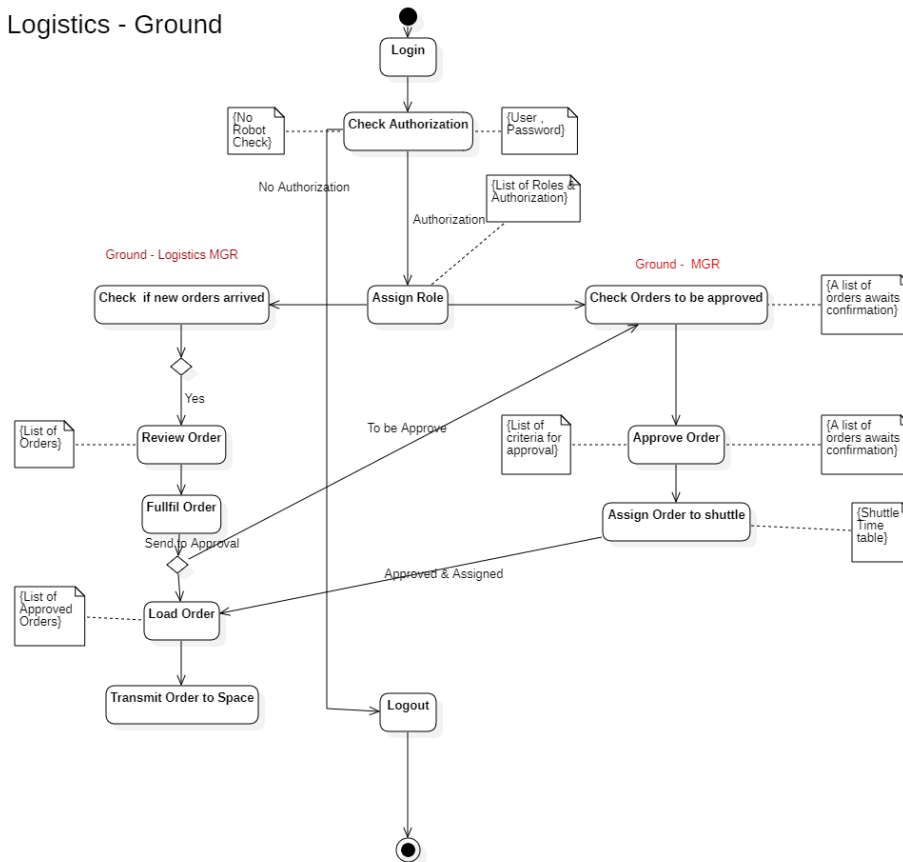
Technical Challenges

Security Challenges

- Secure database - Encrypted data and the use of a database layer that enforces correct usage.
- Encrypted Rest API communications - We can use a public private key approach.
- REST API load monitoring - we monitor the request frequency to the rest Api to try and identify DOS (denial of service) attacks
- Access control- Different levels of access control (could some Astronauts have extended privileges, perhaps access to directly order or view system resources)
- Authentication - Users will need a resisted account and password
- Logging - we could do logging at various layers (the data access layer), perhaps to a text file or database

Architecture and patterns

Our system will be designed to work in a client-server architecture as well as a monolithic architecture. Both designs will make use of a three-tier architecture for separation of concerns. The three tiers will be, interface, business and data storage.

The client-server architecture is designed as such:

Client-Server Architecture
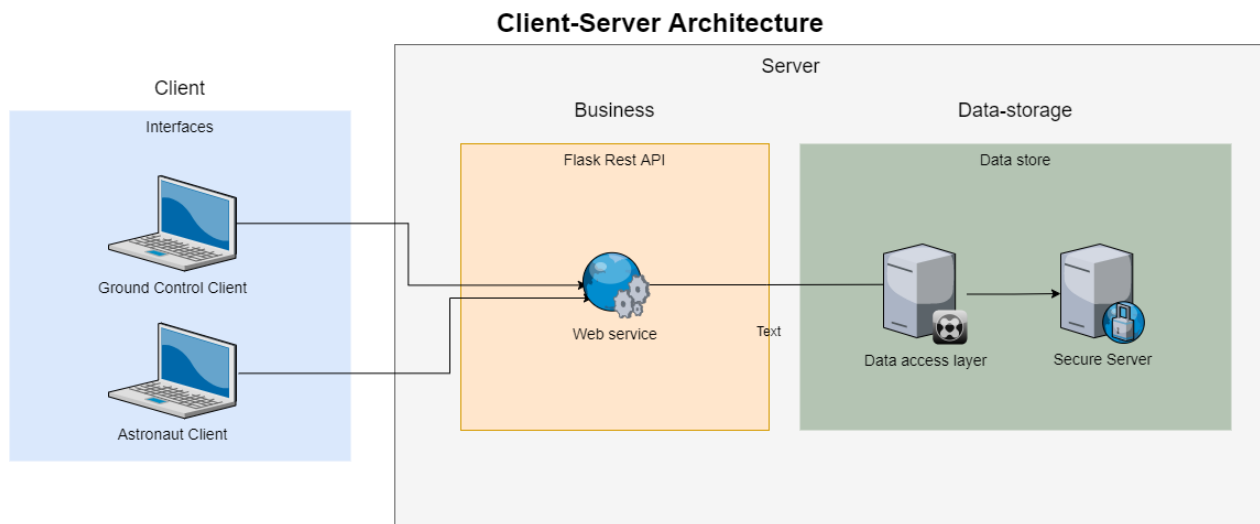
The monolithic architecture will keep the application all part of the same process and will look like:

Monolith Architecture
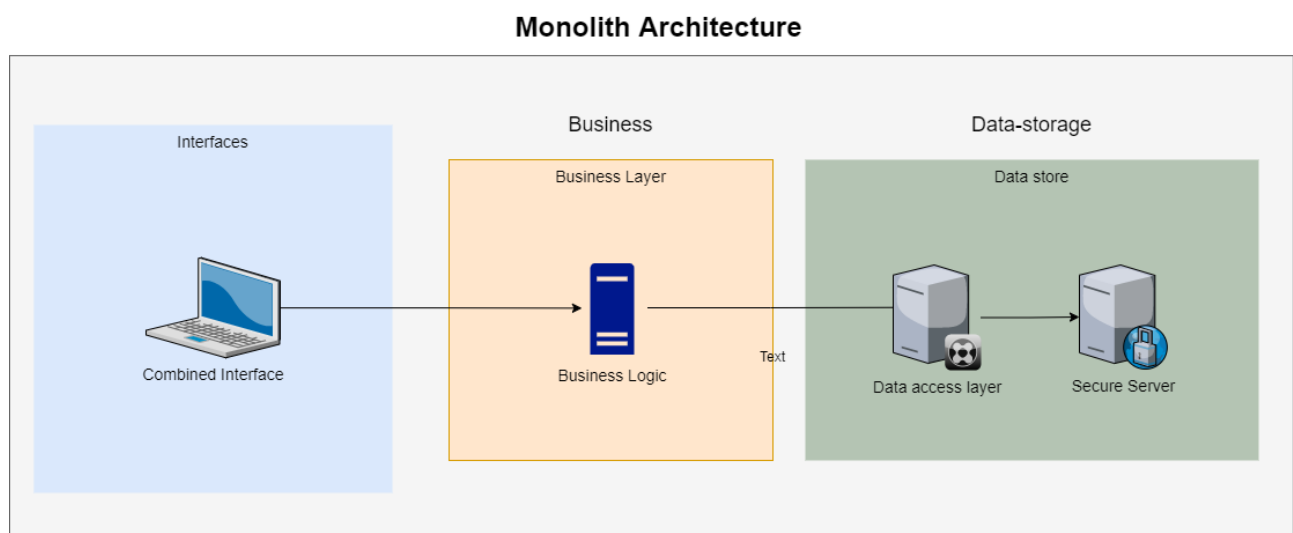
Assumptions and justifications

o Designed according to the specifications of the facilities required by the International Space Station (NASA, 2007).

o Manages critical resources required by the Space station and therefore it has direct effect on the health and security of the team in space.

o Uses the latest methodologies required for safe and secure operation.

o Will be available to be used 24 hours a day, 7 days a week.

o Will make sure that the supply of goods to the space station is to be executed within 72 hours. (24 hours for fulfil the orders
 and 48 hours for journey to space).

-     Testing:

o Performed by independent person with preprepared "Test scenarios" for Unit tests, Integration tests, User's acceptance tests and regression tests.

-      Due to time restrictions not all the facilities needed will be developed. Justifications will be given in the coding. List of technical challenges:

-      Using Phyton as a programing language and SQL database is complicated and required high level of professional conduct.

-      Interfaces between space station and ground control will use secure facilities. Complex encryptions and complex tools.

-      Table logging and monitoring, access control, roles and transactions, passwords maintenance. All combined into one system, creates a whole fabric of business functional and technological challenges.