**API Documentation**

1. RegistrationService.Register(Registration Request): - Path – “api/v1/registration”
   1. Input Json Params: - FirstName, LastName, Password, emailId.
   2. Response: - if email id exists 🡪 User Exists, else 🡪 Token
2. LoginService.Login(Login Request): - Path – api/v1/login
   1. Input Json Params: - UserName (EmailId), Password.
   2. Response: - if match🡪 success, if wrong password🡪 Wrong password, if user\_not\_found🡪 UserName does not exist and Group Name 🡪 to show on home page “welcome to GroupName” top left corner, GroupCode🡪 Group Code on top right corner.
3. GroupServices.CreateGroup(GroupCreation Request): - Path – api/v1/group/create.
   1. Input Json params: - GroupName, GroupCode (Randomly Generated)🡪 also need to pass the user Email Id from Front End.
   2. Response: - “Group Created Successfully’, if Duplicate GroupName🡪 Group Name Exists.
4. GroupServices.JoinGroup(GroupJoinRequest): - Path – api/v1/group/join
   1. Input Json Params: - Group Code🡪 also need to pass the User Email Id from the Front End.
   2. Response: - Joined successfully, else 🡪 please verify the GroupCode: Unable to Join.
5. GroupServices.ViewGroup(ViewGroupRequest): - path – api/v1/group/view.
   1. Input params: - need to pass the UserName (Email Id).
   2. Response: - list of User Details🡪 username, email, phone Number (Optional).
6. GroupServices.leaveGroup(LeaveGroupRequest): - path – api/v1/group/leave.
   1. Input params: - need to pass the UserName (Email Id), group code.
   2. Response: - list of User Details🡪 status , message
7. InventoryServices.AddItem (InventoryAddRequest): - path – api/v1/inventory/add.
   1. Input params: - Item Name, Quantity, Cost, Expiry Date, Category (Selects from drop Down List) and UserName (Email id) and Group Name (If exists).
   2. Response: - Success.
8. InventoryServices.ViewItems(InventoryViewRequest): - api/v1/inventory/view
   1. Input params: - UserName (Email Id).
   2. Response: - List of Items🡪item Name, total Quantity, Expiry Date.
9. InventoryServices.UpdateItem(InventoryViewRequest): - api/v1/inventory/update
   1. Input Params: - Item id, groupName.
   2. Response: - Success.
10. NotificationServices.SendNotification(NotificationSendRequest): - path – api/v1/notification/sendNotification.
    1. Input Params: - item Name, UserName (Email id), Group Name.
    2. Response: - “Notification Sent”, else🡪 no members in group.
11. ExpenseServices.ViewExpenses(ExpenseRequest): - path – api/v1/expense/view.
    1. Input Params: - User Name (Email Id), Group Name.
    2. Response: - list of Usernames 🡪 username, total Spent,
12. GroupExpenseServices.ViewExpenses(ExpenseRequest): - path – api/v1/expense/groupexpenses.
    1. Input Params: - groupnumber.
    2. Response: - list of Usernames 🡪 status,message,groupexpenses
13. groupExpenseServices.totalExpenses(ExpenseRequest): - path – api/v1/expense/total.
14. Input Params: - user name (email id).
15. Response: - list of Usernames 🡪 total expense , status ,message
16. RecipeServices.GetRecipe(RecipeGetRequest): - path – api/v1/recipe/get.
    1. Input Params: - UserName (Email id), group Name (if exists).
    2. Response: - list of recipes.
17. RecipeServices.ViewRecipe(RecipeviewRequest): - path – api/v1/recipe/view
18. Input Params:- group code.
19. 3rd party api used:- https://api.spoonacular.com/recipes/findByIngredients

https://api.spoonacular.com/recipes

1. Response: recipes
2. NotificationServices.NotificationRequest(NotificationGetRequest) – path – api/v1/notification/notify
   1. Input Params: - group code , item name.
   2. Response: - list of Usernames 🡪 total expense , status ,message